

INTERNET-DRAFT  
Intended Status: Informational  
Expires: August 12, 2014

A. Ghanwani  
Dell  
L. Dunbar  
Huawei  
V. Bannai  
Paypal  
R. Krishnan  
Brocade  
February 13, 2014

Multicast Issues in Networks Using NVO3  
draft-ghanwani-nvo3-mcast-issues-01

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at  
<http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at  
<http://www.ietf.org/shadow.html>

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document.

## Abstract

This memo discusses issues with supporting multicast traffic in a network that uses Network Virtualization using Overlays over Layer 3 (NVO3). It describes the various mechanisms that may be used for multicast and discusses some of the considerations with supporting multicast applications in networks that use NVO3.

## Table of Contents

1. Introduction . . . . .	4
2. Multicast mechanisms in networks that use NVO3 . . . . .	4
2.1 No multicast support . . . . .	4
2.2 Replication at the source NVE . . . . .	5
2.3 Replication at a multicast service node . . . . .	5
2.4 IP multicast in the underlay . . . . .	6
2.5 Other schemes . . . . .	7
3. Simultaneous use of more than one mechanism . . . . .	7
4. IP multicast applications in the overlay . . . . .	7
5. Summary . . . . .	8
6. Security Considerations . . . . .	8
7. IANA Considerations . . . . .	8
8. References . . . . .	8
8.1 Normative References . . . . .	8
8.2 Informative References . . . . .	9
Authors' Addresses . . . . .	9

## 1. Introduction

Network virtualization using Overlays over Layer 3 (NVO3) is a technology that is used to address issues that arise in building large, multitenant data centers that make extensive use of server virtualization [PS].

This document is focused specifically on the problem of supporting multicast in networks that use NVO3. Because of the requirement of multi-destination delivery, multicast traffic poses some unique challenges.

The reader is assumed to be familiar with the terminology as defined in the NVO3 Framework document [FW].

## 2. Multicast mechanisms in networks that use NVO3

In NVO3 environments, traffic between NVEs is transported using a tunnel encapsulation such as VXLAN [VXLAN], NVGRE [NVGRE], STT [STT], etc.

Besides the need to support the Address Resolution Protocol (ARP) and Neighbor Discovery (ND), there are several applications that require the support of multicast and/or broadcast in data centers [DC-MC]. With NVO3, there are many possible ways that multicast may be handled in such networks. We discuss some of the attributes of the following four methods, but other methods are also possible.

1. No multicast support.
2. Replication at the source NVE.
3. Replication at a multicast service node.
4. IP multicast in the underlay.

These mechanisms are briefly mentioned in the NVO3 Framework [FW] document. This document attempts to fill in some more details about the basic mechanisms underlying each of these mechanisms and discusses the issues and tradeoffs of each.

### 2.1 No multicast support

In this scenario, there is no support whatsoever for multicast traffic when using the overlay. This can only work if the following conditions are met:

1. All of the traffic is unicast. In other words, there are no multicast applications in the network and the only multicast traffic is due to ARP/ND and due to flooding of frames with an unknown MAC destination address.

2. A network virtualization authority (NVA) is used at the NVE to determine the MAC address-to-NVE mapping and to determine the MAC address-to-IP address bindings. In other words, there is no data plane learning, and address resolution requests via ARP/ND that are issued by the VMs must be resolved by the NVE that they are attached to.

With this approach, certain multicast/broadcast applications such as DHCP can be supported by use of a helper function in the NVE.

The main issues that need to be addressed with this mechanism are the handling of hosts for which a mapping does not already exist in the NVA. This issue can be particularly challenging if such end systems are reachable through more than one NVE.

## 2.2 Replication at the source NVE

With this method, the overlay attempts to provide a multicast service without requiring any specific support from the underlay, other than that of a unicast service. A multicast or broadcast transmission is achieved by replicating the packet at the source NVE, and making copies, one for each destination NVE that the multicast packet must be sent to.

For this mechanism to work, the source NVE must know, a priori, the IP addresses of all destination NVEs that need to receive the packet.

For example, in the case of an ARP broadcast or an ND multicast, the source NVE must know the IP addresses of all the remote NVEs where there are members of the tenant subnet in question.

The obvious drawback with this method is that we have multiple copies of the same packet that will traverse any common links that are along the path to each of the destination NVEs. If, for example, a tenant subnet is spread across 50 NVEs, the packet would have to be replicated 50 times at the source NVE. This also creates an issue with the forwarding performance of the NVE, especially if it is implemented in software.

Note that this method is similar to what was used in VPLS [VPLS] prior to extensive support of MPLS multicast [MPLS-MC].

## 2.3 Replication at a multicast service node

With this method, all multicast packets would be sent using a unicast tunnel encapsulation to a multicast service node. The multicast service node, in turn, would create multiple copies of the packet and would deliver a copy, using a unicast tunnel encapsulation, to each of the NVEs that are part of the multicast group for which the packet

is intended.

This mechanism is similar to that used by the ATM Forum's LAN Emulation [LANE] specification [LANE].

Unlike the method described in Section 2.2, there is no performance impact at the ingress NVE, nor are there any issues with multiple copies of the same packet from the source NVE to the multicast service node. However there remain issues with multiple copies of the same packet on links that are common to the paths from the multicast service node to each of the egress NVEs. Additional issues that are introduced with this method include the availability of the multicast service node, methods to scale the services offered by the multicast service node, and the sub-optimality of the delivery paths.

Finally, the IP address of the source NVE must be preserved in packet copies created at the multicast service node if data plane learning is in use. This could create problems if IP source address reverse path forwarding (RPF) checks are in use.

#### 2.4 IP multicast in the underlay

In this method, the underlay supports IP multicast and the ingress NVE encapsulates the packet with the appropriate IP multicast address in the tunnel encapsulation header for delivery to the desired set of NVEs. The protocol in the underlay could be any variant of Protocol Independent Multicast (PIM). The NVE would be required to participate in the underlay as a host using IGMP/MLD in order for the underlay to learn about the groups that the NVE participates in.

With this method, there are none of the issues with the methods described in Sections 2.2.

With PIM Sparse Mode (PIM-SM), the number of flows required would be  $(n \cdot g)$ , where  $n$  is the number of source NVEs that source packets for the group, and  $g$  is the number of groups. Bidirectional PIM (BIDIR-PIM) would offer better scalability with the number of flows required being  $g$ .

In the absence of any additional mechanism, e.g. using an NVA for address resolution, for optimal delivery, there would have to be a separate group for each tenant, plus a separate group for each multicast address (used for multicast applications) within a tenant. Additional considerations are that only the lower 23 bits of the IP address (regardless of whether IPv4 or IPv6 is in use) are mapped to the outer MAC address, and if there is equipment that prunes multicasts at Layer 2, there will be some aliasing. Finally, a mechanism to efficiently provision such addresses for each group

would be required.

There are additional optimizations which are possible, but they come with their own restrictions. For example, a set of tenants may be restricted to some subset of NVEs and they could all share the same outer IP multicast group address. This however introduces a problem of sub-optimal delivery (even if a particular tenant within the group of tenants doesn't have a presence on one of the NVEs which another one does, the former's multicast packets would still be delivered to that NVE). It also introduces an additional network management burden to optimize which tenants should be part of the same tenant group (based on the NVEs they share), which somewhat dilutes the value proposition of NVO3 which is to completely decouple the overlay and physical network design allowing complete freedom of placement of VMs anywhere within the data center.

## 2.5 Other schemes

There are still other mechanisms that may be used that attempt to combine some of the advantages of the above methods by offering multiple replication points, each with a limited degree of replication [EDGE-REP]. Such schemes offer a trade-off between the amount of replication at an intermediate node (router) versus performing all of the replication at the source NVE or all of the replication at a multicast service node.

## 3. Simultaneous use of more than one mechanism

While the mechanisms discussed in the previous section have been discussed individually, it is possible for implementations to rely on more than one of these. For example, the method of Section 2.1 could be used for minimizing ARP/ND, while at the same time, multicast applications may be supported by one, or a combination of, the other methods. For small multicast groups, the methods of source NVE replication or the use of a multicast service node may be attractive, while for larger multicast groups, the use of multicast in the underlay may be preferable.

## 4. IP multicast applications in the overlay

When IP multicast is implemented in the overlay (i.e. the tenant traffic is IP multicast), there are a few issues that need to be addressed.

First, in all cases where L2 virtual network interfaces (VNIs) are present, the NVE would need to support IGMP/MLD snooping in order to prevent delivery of packets to tenant systems that are not interested in receiving them.

Second is the issue of how the groups are setup and mapped to tunnels in the underlay. This can be accomplished entirely by an NVA if the mechanisms described in Section 2.2 or Section 2.3 are used, with the NVE just participating in snooping of IGMP messages from the tenant systems. If the method of Section 2.4 is used, then a mechanism must be provide for mapping the tenant IP multicast address to an IP multicast address for use in the underlay, and the NVE would be required to translate the information from the snooped IGMP/MLD messages from the tenant systems into corresponding requests for the underlay.

Third, when using the scheme described in Section 2.3, it may be useful to have the multicast service node support the IGMP querier function.

Fourth, if the IP multicast traffic is contained within a single virtual network (VN), then the schemes described herein are sufficient. If, on the other hand, the IP multicast traffic needs to traverse VNs, then the routing mechanisms at the NVE need to offer IP multicast forwarding. Once again, depending on how the groups are setup -- whether by an NVA or some other entity -- the forwarding tables at the NVE that has L3 virtual network interfaces (VNIs) would need to be setup by that entity.

## 5. Summary

This document has identified various mechanisms for supporting multicast in networks that use NVO3. It highlights the basics of each mechanism and some of the issues with them. As solutions are developed, the protocols would need to consider the use of these mechanisms and co-existence may be a consideration. It also highlights some of the requirements for supporting multicast applications in an NVO3 network.

## 6. Security Considerations

This is an informational document, and as such, does not introduce any new security considerations beyond what may be present in proposed solutions.

## 7. IANA Considerations

This draft does not have any IANA considerations.

## 8. References

### 8.1 Normative References



- [PS] Lasserre, M. et al., "Framework for DC network virtualization", work in progress, January 2014.
- [FW] Narten, T. et al., "Problem statement: Overlays for network virtualization", work in progress, July 2013.

## 8.2 Informative References

- [VXLAN] Mahalingam, M. et al., "VXLAN: A framework for overlaying virtualized Layer 2 networks over Layer 3 networks," work in progress.
- [NVGRE] Sridharan, M. et al., "NVGRE: Network virtualization using Generic Routing Encapsulation," work in progress.
- [STT] Davie, B. and Gross J., "A stateless transport tunneling protocol for network virtualization," work in progress.
- [DC-MC] McBride M., and Lui, H., "Multicast in the data center overview," work in progress.
- [VPLS] Lasserre, M., and Kompella, V. (Eds), "Virtual Private LAN Service (VPLS) using Label Distribution Protocol (LDP) signaling," RFC 4762, January 2007.
- [MPLS-MC] Aggarwal, R. et al., "Multicast in VPLS," work in progress.
- [LANE] "LAN emulation over ATM," The ATM Forum, af-lane-0021.000, January 1995.
- [EDGE-REP] Marques P. et al., "Edge multicast replication for BGP IP VPNs," work in progress, June 2012.

## Authors' Addresses

Anoop Ghanwani  
Dell  
Email: [anoop@alumni.duke.edu](mailto:anoop@alumni.duke.edu)

Linda Dunbar  
Huawei  
Email: [ldunbar@huawei.com](mailto:ldunbar@huawei.com)

Vinay Bannai  
Paypal  
Email: vbannai@paypal.com

Ram Krishnan  
Brocade  
Email: ramk@brocade.com

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: August 18, 2014

J. Gross  
T. Sridhar  
VMware  
P. Garg  
Microsoft  
C. Wright  
Red Hat  
I. Ganga  
Intel  
February 14, 2014

Geneve: Generic Network Virtualization Encapsulation  
draft-gross-geneve-00

Abstract

Network virtualization involves the cooperation of devices with a wide variety of capabilities such as software and hardware tunnel endpoints, transit fabrics, and centralized control clusters. As a result of their role in tying together different elements in the system, the requirements on tunnels are influenced by all of these components. Flexibility is therefore the most important aspect of a tunnel protocol if it is keep pace with the evolution of the system. This draft describes Geneve, a protocol designed to recognize and accommodate these changing capabilities and needs.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 18, 2014.

## Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

## Table of Contents

1. Introduction . . . . .	3
1.1. Requirements Language . . . . .	4
1.2. Terminology . . . . .	4
2. Design Requirements . . . . .	5
2.1. Control Plane Independence . . . . .	6
2.2. Efficient Implementation . . . . .	7
2.3. Use of Standard IP Fabrics . . . . .	8
3. Geneve Encapsulation Details . . . . .	8
3.1. Geneve Frame Format Over IPv4 . . . . .	9
3.2. Geneve Frame Format Over IPv6 . . . . .	10
3.3. UDP Header . . . . .	12
3.4. Tunnel Header Fields . . . . .	13
3.5. Tunnel Options . . . . .	14
3.5.1. Options Processing . . . . .	16
4. Implementation and Deployment Considerations . . . . .	16
4.1. Encapsulation of Geneve in IP . . . . .	16
4.1.1. IP Fragmentation . . . . .	16
4.1.2. DSCP and ECN . . . . .	17
4.1.3. Broadcast and Multicast . . . . .	17
4.2. NIC Offloads . . . . .	18
4.3. Inner VLAN Handling . . . . .	19
5. Interoperability Issues . . . . .	19
6. Security Considerations . . . . .	20
7. IANA Considerations . . . . .	20
8. Acknowledgements . . . . .	20
9. References . . . . .	20
9.1. Normative References . . . . .	21
9.2. Informative References . . . . .	21
Authors' Addresses . . . . .	22

## 1. Introduction

Networking has long featured a variety of tunneling, tagging, and other encapsulation mechanisms. However, the advent of network virtualization has caused a surge of renewed interest and a corresponding increase in the introduction of new protocols. The large number of protocols in this space, ranging all the way from VLANs [IEEE.802.1Q-2011] and MPLS [RFC3031] through the more recent VXLAN [I-D.mahalingam-dutt-dcops-vxlan], NVGRE [I-D.sridharan-virtualization-nvgre], and STT [I-D.davie-stt], often leads to questions about the need for new encapsulation formats and what it is about network virtualization in particular that leads to their proliferation.

While many encapsulation protocols seek to simply partition the underlay network or bridge between two domains, network virtualization views the transit network as providing connectivity between multiple components of an integrated system. In many ways this system is similar to a chassis switch with the IP underlay network playing the role of the backplane and tunnel endpoints on the edge as line cards. When viewed in this light, the requirements placed on the tunnel protocol are significantly different in terms of the quantity of metadata necessary and the role of transit nodes.

Current work such as [VL2] and the NVO3 working group [I-D.ietf-nvo3-dataplane-requirements] have described some of the properties that the data plane must have to support network virtualization. However, one additional defining requirement is the need to carry system state along with the packet data. The use of some metadata is certainly not a foreign concept - nearly all protocols used for virtualization have at least 24 bits of identifier space as a way to partition between tenants. This is often described as overcoming the limits of 12-bit VLANs, and when seen in that context, or any context where it is a true tenant identifier, 16 million possible entries is a large number. However, the reality is that the metadata is not exclusively used to identify tenants and encoding other information quickly starts to crowd the space. In fact, when compared to the tags used to exchange metadata between line cards on a chassis switch, 24-bit identifiers start to look quite small. There are nearly endless uses for this metadata, ranging from storing input ports for simple security policies to service based context for interposing advanced middleboxes.

Existing tunnel protocols have each attempted to solve different aspects of these new requirements, only to be quickly rendered out of date by changing control plane implementations and advancements. Furthermore, software and hardware components and controllers all have different advantages and rates of evolution - a fact that should

be viewed as a benefit, not a liability or limitation. This draft describes Geneve, a protocol which seeks to avoid these problems by providing a framework for tunneling rather than being prescriptive about the entire system.

### 1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying RFC-2119 significance.

### 1.2. Terminology

The following terms are used in this document:

Checksum offload. An optimization implemented by many NICs which enables computation and verification of upper layer protocol checksums in hardware on transmit and receive, respectively. This typically includes IP and TCP/UDP checksums which would otherwise be computed by the protocol stack in software.

Clos network. A technique for composing network fabrics larger than a single switch while maintaining non-blocking bandwidth across connection points. ECMP is used to divide traffic across the multiple links and switches that constitute the fabric. Sometimes termed "leaf and spine" or "fat tree" topologies.

ECMP. Equal Cost Multipath. A routing mechanism for selecting from among multiple best next hop paths by hashing packet headers in order to better utilize network bandwidth while avoiding reordering a single stream.

Geneve. Generic Network Virtualization Encapsulation. The tunnel protocol described in this draft.

LRO. Large Receive Offload. The receive-side equivalent function of LSO, in which multiple protocol segments (primarily TCP) are coalesced into larger data units.

NIC. Network Interface Card. A NIC could be part of a tunnel endpoint or transit device and can either process Geneve packets or aid in the processing of Geneve packets.

OAM. Operations, Administration, and Management. A suite of tools used to monitor and troubleshoot network problems.

Transit device. A forwarding element along the path of the tunnel. A transit device MAY be capable of understanding the Geneve frame format but does not originate or terminate Geneve packets.

LSO. Large Segmentation Offload. A function provided by many commercial NICs that allows data units larger than the MTU to be passed to the NIC to improve performance, the NIC being responsible for creating smaller segments with correct protocol headers. When referring specifically to TCP/IP, this feature is often known as TSO (TCP Segmentation Offload).

Tunnel endpoint. A component encapsulating Ethernet frames in Geneve headers and vice versa. As the ultimate consumer of any tunnel metadata, endpoints have the highest level of requirements for parsing and interpreting tunnel headers. Tunnel endpoints may consist of either software or hardware implementations or a combination of the two.

VM. Virtual Machine.

## 2. Design Requirements

Geneve is designed to support network virtualization use cases, where tunnels are typically established to act as a backplane between the virtual switches residing in hypervisors, physical switches, or middleboxes or other appliances. An arbitrary IP network can be used as an underlay although Clos networks composed using ECMP links are a common choice to provide consistent bisectional bandwidth across all connection points. Figure 1 shows an example of a hypervisor, top of rack switch for connectivity to physical servers, and a WAN uplink connected using Geneve tunnels over a simplified Clos network. These tunnels are used to encapsulate and forward frames from the attached components such as VMs or physical links.

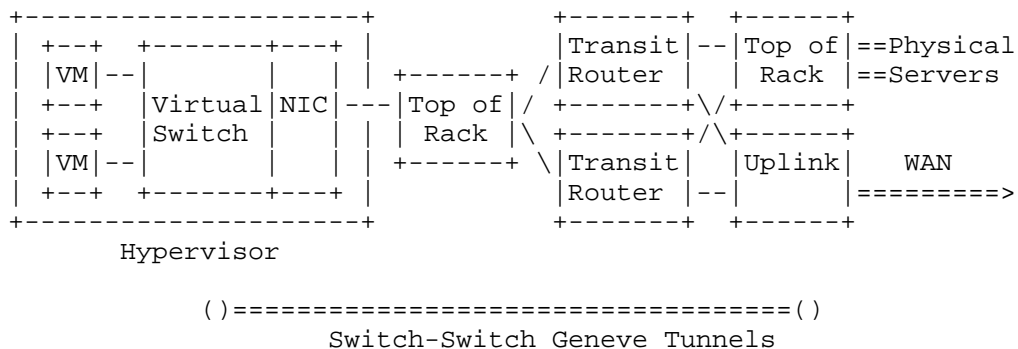


Figure 1: Sample Geneve Deployment

In order to support the needs of network virtualization, the tunnel protocol should be able to take advantage of the differing (and evolving) capabilities of each type of device in both the underlay and overlay networks. This results in the following requirements being placed on the data plane tunneling protocol:

- o The data plane is generic and extensible enough to support current and future control planes.
- o Tunnel components are efficiently implementable in both hardware and software without restricting capabilities to the lowest common denominator.
- o High performance over existing IP fabrics.

These requirements are described further in the following subsections.

## 2.1. Control Plane Independence

Although some protocols for network virtualization have included a control plane as part of the tunnel format specification (most notably, the original VXLAN spec prescribed a multicast learning-based control plane), these specifications have largely been treated as describing only the data format. The VXLAN frame format has actually seen a wide variety of control planes built on top of it.

There is a clear advantage in settling on a data format: most of the protocols are only superficially different and there is little advantage in duplicating effort. However, the same cannot be said of control planes, which are diverse in very fundamental ways. The case for standardization is also less clear given the wide variety in requirements, goals, and deployment scenarios.



As a result of this reality, Geneve aims to be a pure tunnel format specification that is capable of fulfilling the needs of many control planes by explicitly not selecting any one of them. This simultaneously promotes a shared data format and increases the chances that it will not be obsoleted by future control plane enhancements.

Achieving this level of flexibility effectively requires an options infrastructure to allow new metadata types to be defined, deployed, and either finalized or retired. Options also allow for differentiation of products by encouraging independent development in each vendor's core specialty, leading to an overall faster pace of advancement. By far the most common mechanism for implementing options is Type-Length-Value (TLV) format.

It should be noted that while options can be used to support non-wirespeed control frames, they are equally important on data frames as well to segregate and direct forwarding (for instance, the examples given before of input port based security policies and service interposition both require tags to be placed on data packets). Therefore, while it would be desirable to limit the extensibility to only control frames for the purposes of simplifying the datapath, that would not satisfy the design requirements.

## 2.2. Efficient Implementation

There is often a conflict between software flexibility and hardware performance that is difficult to resolve. For a given set of functionality, it is obviously desirable to maximize performance. However, that does not mean new features that cannot be run at that speed today should be disallowed. Therefore, for a protocol to be efficiently implementable means that a set of common capabilities can be reasonably handled across platforms along with a graceful mechanism to handle more advanced features in the appropriate situations.

The use of a variable length header and options in a protocol often raises questions about whether it is truly efficiently implementable in hardware. To answer this question in the context of Geneve, it is important to first divide "hardware" into two categories: tunnel endpoints and transit devices.

Endpoints must be able to parse the variable header, including any options, and take action. Since these devices are actively participating in the protocol, they are the most affected by Geneve. However, as endpoints are the ultimate consumers of the data, transmitters can tailor their output to the capabilities of the recipient. As new functionality becomes sufficiently well defined to

add to endpoints, supporting options can be designed using ordering restrictions and other techniques to ease parsing.

Transit devices MAY be able to interpret the options and participate in Geneve packet processing. However, as non-terminating devices, they do not originate or terminate the Geneve packet. The participation of transit devices in Geneve packet processing is OPTIONAL.

Further, either tunnel endpoints or transit devices MAY use offload capabilities of NICs such as checksum offload to improve the performance of Geneve packet processing. The presence of a Geneve variable length header SHOULD NOT prevent the tunnel endpoints and transit devices from using such offload capabilities.

### 2.3. Use of Standard IP Fabrics

IP has clearly cemented its place as the dominant transport mechanism and many techniques have evolved over time to make it robust, efficient, and inexpensive. As a result, it is natural to use IP fabrics as a transit network for Geneve. Fortunately, the use of IP encapsulation and addressing is enough to achieve the primary goal of delivering packets to the correct point in the network through standard switching and routing.

In addition, nearly all underlay fabrics are designed to exploit parallelism in traffic to spread load across multiple links without introducing reordering in individual flows. These equal cost multipathing (ECMP) techniques typically involve parsing and hashing the addresses and port numbers from the packet to select an outgoing link. However, the use of tunnels often results in poor ECMP performance without additional knowledge of the protocol as the encapsulated traffic is hidden from the fabric by design and only endpoint addresses are available for hashing.

Since it is desirable for Geneve to perform well on these existing fabrics, it is necessary for entropy from encapsulated packets to be exposed in the tunnel header. The most common technique for this is to use the UDP source port, which is discussed further in Section 3.3.

### 3. Geneve Encapsulation Details

The Geneve frame format consists of a compact tunnel header encapsulated in UDP over either IPv4 or IPv6. A small fixed tunnel header provides control information plus a base level of functionality and interoperability with a focus on simplicity. This header is then followed by a set of variable options to allow for

future innovation. Finally, the payload consists of a protocol data unit of the indicated type, such as an Ethernet frame. The following subsections provide examples of Geneve frames transported (for example) over Ethernet along with an Ethernet payload.

### 3.1. Geneve Frame Format Over IPv4

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

```

Outer Ethernet Header:

```

+-----+
|                               Outer Destination MAC Address                               |
+-----+
| Outer Destination MAC Address | Outer Source MAC Address |
+-----+
|                               Outer Source MAC Address                               |
+-----+
| Optional Ethertype=C-Tag 802.1Q | Outer VLAN Tag Information |
+-----+
|                               Ethertype=0x0800                               |
+-----+

```

Outer IPv4 Header:

```

+-----+
| Version | IHL | Type of Service | Total Length |
+-----+
| Identification | Flags | Fragment Offset |
+-----+
| Time to Live | Protocol=17 UDP | Header Checksum |
+-----+
|                               Outer Source IPv4 Address                               |
+-----+
|                               Outer Destination IPv4 Address                               |
+-----+

```

Outer UDP Header:

```

+-----+
| Source Port = xxxx | Dest Port = Geneve Port |
+-----+
| UDP Length | UDP Checksum |
+-----+

```

## Geneve Header:

```

+-----+
|Ver|  Opt Len  |O|C|    Rsvd.   |          Protocol Type          |
+-----+
|          Virtual Network Identifier (VNI)          |      Reserved      |
+-----+
|          Variable Length Options          |
+-----+

```

## Inner Ethernet Header:

```

+-----+
|          Inner Destination MAC Address          |
+-----+
| Inner Destination MAC Address | Inner Source MAC Address |
+-----+
|          Inner Source MAC Address          |
+-----+
|Optional Ethertype=C-Tag 802.1Q| Inner VLAN Tag Information |
+-----+

```

## Payload:

```

+-----+
| Ethertype of Original Payload |
+-----+
|          Original Ethernet Payload          |
| (Note that the original Ethernet Frame's FCS is not included) |
+-----+

```

## Frame Check Sequence:

```

+-----+
| New FCS (Frame Check Sequence) for Outer Ethernet Frame |
+-----+

```

## 3.2. Geneve Frame Format Over IPv6

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

```

## Outer Ethernet Header:

```

+-----+
|                               Outer Destination MAC Address                               |
+-----+
| Outer Destination MAC Address | Outer Source MAC Address |
+-----+
|                               Outer Source MAC Address                               |
+-----+
| Optional Ethertype=C-Tag 802.1Q | Outer VLAN Tag Information |
+-----+
|                               Ethertype=0x86DD                               |
+-----+

```

## Outer IPv6 Header:

```

+-----+
| Version | Traffic Class |                               Flow Label                               |
+-----+
|                               Payload Length                               | NxtHdr=17 UDP | Hop Limit |
+-----+
|
+
|                               Outer Source IPv6 Address                               |
+
|
+-----+
|
+
|                               Outer Destination IPv6 Address                               |
+
|
+-----+

```

## Outer UDP Header:

```

+-----+
|                               Source Port = xxxx                               | Dest Port = Geneve Port |
+-----+
|                               UDP Length                               | UDP Checksum |
+-----+

```

## Geneve Header:

```

+-----+
|Ver|  Opt Len  |O|C|    Rsvd.   |          Protocol Type          |
+-----+
|          Virtual Network Identifier (VNI)          |      Reserved      |
+-----+
|          Variable Length Options          |
+-----+

```

## Inner Ethernet Header:

```

+-----+
|          Inner Destination MAC Address          |
+-----+
| Inner Destination MAC Address | Inner Source MAC Address |
+-----+
|          Inner Source MAC Address          |
+-----+
|Optional Ethertype=C-Tag 802.1Q| Inner VLAN Tag Information |
+-----+

```

## Payload:

```

+-----+
| Ethertype of Original Payload |
+-----+
|          Original Ethernet Payload          |
|
| (Note that the original Ethernet Frame's FCS is not included) |
+-----+

```

## Frame Check Sequence:

```

+-----+
| New FCS (Frame Check Sequence) for Outer Ethernet Frame |
+-----+

```

## 3.3. UDP Header

The use of an encapsulating UDP [RFC0768] header follows the connectionless semantics of Ethernet and IP in addition to providing entropy to routers performing ECMP. The header fields are therefore interpreted as follows:

**Source port:** A source port selected by the ingress tunnel endpoint. This source port SHOULD be the same for all packets belonging to a single encapsulated flow to prevent reordering due to the use of different paths. To encourage an even distribution of flows across multiple links, the source port MAY be calculated using a hash of the encapsulated packet headers using, for example, a traditional 5-tuple. Since the port represents a flow identifier

rather than a true UDP connection, the entire 16-bit range MAY be used to maximize entropy.

Dest port: Fixed well-known destination port to be allocated by IANA. This port MUST be used in both directions of a flow. Since a port has not yet been assigned, it is RECOMMENDED that implementations make this configurable.

UDP length: The length of the UDP packet including the UDP header.

UDP checksum: The checksum MAY be set to zero on transmit for packets encapsulated in both IPv4 and IPv6 [RFC6935]. When a packet is received with a UDP checksum of zero it MUST be accepted and decapsulated. If the ingress tunnel endpoint optionally encapsulates a packet with a non-zero checksum, it MUST be a correctly computed UDP checksum. Upon receiving such a packet, the egress endpoint MAY validate the checksum. If the receiver chooses to perform verification and the checksum is not correct, the packet MUST be dropped. Otherwise, the packet MUST be accepted for decapsulation. It is RECOMMENDED that the UDP checksum be computed to protect the Geneve header and options in situations where the network reliability is not high and the packet is not protected by another checksum or CRC.

### 3.4. Tunnel Header Fields

Ver (2 bits): The current version number is 0. Packets received by an endpoint with an unknown version MUST be dropped. Non-terminating devices processing Geneve packets with an unknown version number MUST treat them as UDP packets with an unknown payload.

Opt Len (6 bits): The length of the options fields, expressed in four byte multiples, not including the eight byte fixed tunnel header. This results in a minimum total Geneve header size of 8 bytes and a maximum of 260 bytes. The start of the payload headers can be found using this offset from the end of the base Geneve header.

O (1 bit): OAM frame. This packet contains a control message instead of a data payload. Endpoints MUST NOT forward the payload and transit devices MUST NOT attempt to interpret or process it. Since these are infrequent control messages, it is RECOMMENDED that endpoints direct these packets to a high priority control queue (for example, to direct the packet to a general purpose CPU from a forwarding ASIC or to separate out control traffic on a NIC). Transit devices MUST NOT alter forwarding behavior on the basis of this bit, such as ECMP link selection.

C (1 bit): Critical options present. One or more options has the critical bit set (see Section 3.5). If this bit is set then tunnel endpoints MUST parse the options list to interpret any critical options. If no option types are supported then endpoints MAY silently drop the frame on the basis of the 'C' bit (including invalid combinations such as 'C' bit set and 'Opt Len' is zero or no options with a corresponding 'C' bit). If the bit is not set tunnel endpoints MAY strip all options using 'Opt Len' and forward the decapsulated frame. Transit devices MUST NOT drop or modify packets on the basis of this bit.

Rsvd. (6 bits): Reserved field which MUST be zero on transmission and ignored on receipt.

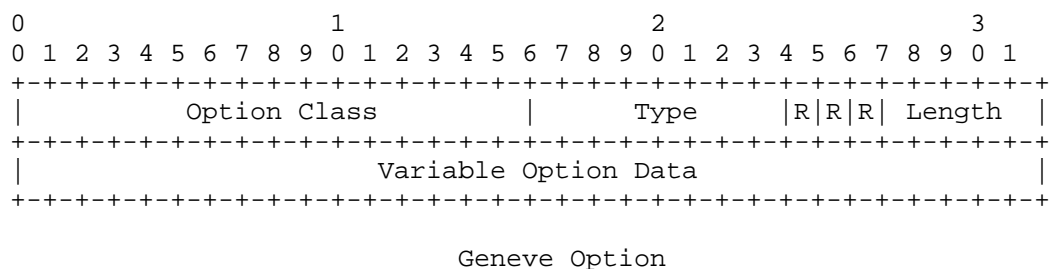
Protocol Type (16 bits): The type of the protocol data unit appearing after the Geneve header. This follows the EtherType [ETYPES] convention with Ethernet itself being represented by the value 0x6558.

Virtual Network Identifier (VNI) (24 bits): An identifier for a unique element of a virtual network. In many situations this may represent an L2 segment, however, the control plane defines the forwarding semantics of decapsulated packets. The VNI MAY be used as part of ECMP forwarding decisions or MAY be used as a mechanism to distinguish between overlapping address spaces contained in the encapsulated packet when load balancing across CPUs.

Reserved (8 bits): Reserved field which MUST be zero on transmission and ignored on receipt.

Transit devices MUST maintain consistent forwarding behavior irrespective of the value of 'Opt Len', including ECMP link selection. These devices SHOULD be able to forward packets containing options without resorting to a slow path.

### 3.5. Tunnel Options





The base Geneve header is followed by zero or more options in Type-Length-Value format. Each option consists of a four byte option header and a variable amount of option data interpreted according to the type.

Option Class (16 bits): Namespace for the 'Type' field. IANA will be requested to create a "Geneve Option Class" registry to allocate identifiers for organizations, technologies, and vendors that have an interest in creating types for options. Each organization may allocate types independently to allow experimentation and rapid innovation. It is expected that over time certain options will become well known and a given implementation may use option types from a variety of sources. In addition, IANA will be requested to reserve specific ranges for standardized and experimental options.

Type (8 bits): Type indicating the format of the data contained in this option. Options are primarily designed to encourage future extensibility and innovation and so standardized forms of these options will be defined in a separate document.

The high order bit of the option type indicates that this is a critical option. If the receiving endpoint does not recognize this option and this bit is set then the frame MUST be dropped. If the critical bit is set in any option then the 'C' bit in the Geneve base header MUST also be set. Transit devices MUST NOT drop packets on the basis of this bit.

The requirement to drop a packet with an unknown critical option applies to the entire tunnel endpoint system and not a particular component of the implementation. For example, in a system comprised of a forwarding ASIC and a general purpose CPU, this does not mean that the packet must be dropped in the ASIC. An implementation may send the packet to the CPU using a rate-limited control channel for slow-path exception handling.

R (1 bit): Option control flag reserved for future use. MUST be zero on transmission and ignored on receipt.

Length (5 bits): Length of the option, expressed in four byte multiples excluding the option header. The total length of each option may be between 4 and 128 bytes. Packets in which the total length of all options is not equal to the 'Opt Len' in the base header are invalid and MUST be silently dropped if received by an endpoint.

Variable Option Data: Option data interpreted according to 'Type'.

### 3.5.1. Options Processing

Geneve options are intended to be originated and processed by tunnel endpoints. Options MAY be processed by transit devices along the tunnel path as well. This document only details the handling of options by tunnel endpoints. A future version of this document will provide details of options processing by transit devices. Transit devices not processing Geneve options SHOULD process Geneve frame as any other UDP frame and maintain consistent forwarding behavior.

In tunnel endpoints, the generation and interpretation of options is determined by the control plane, which is out of the scope of this document. However, to ensure interoperability between heterogeneous devices two requirements are imposed on endpoint devices:

- o Receiving endpoints MUST drop packets containing unknown options with the 'C' bit set in the option type.
- o Sending endpoints MUST NOT assume that options will be processed sequentially by the receiver in the order they were transmitted.

## 4. Implementation and Deployment Considerations

### 4.1. Encapsulation of Geneve in IP

As an IP-based tunnel protocol, Geneve shares many properties and techniques with existing protocols. The application of some of these are described in further detail, although in general most concepts applicable to the IP layer or to IP tunnels generally also function in the context of Geneve.

#### 4.1.1. IP Fragmentation

In order to prevent fragmentation and maximize performance, the best practice when using Geneve is to ensure that the MTU of the physical network is greater than or equal to the MTU of the encapsulated network plus tunnel headers. Manual or upper layer (such as TCP MSS clamping) configuration can be used to ensure that fragmentation never takes place, however, in some situations this may not be feasible.

It is RECOMMENDED that Path MTU Discovery be used by setting the DF bit in the IP header when Geneve packets are transmitted over IPv4 (this is the default with IPv6). The use of Path MTU Discovery on the transit network provides the encapsulating endpoint with soft-state about the link that it may use to prevent or minimize fragmentation depending on its role in the virtualized network.

If necessary, it is RECOMMENDED that fragmentation be performed preferentially on the encapsulated payload. This may be possible if the encapsulating endpoint is also acting as an L3 node in the virtualized network, in which case the endpoint might use the derived transit MTU and the tunnel header length to either implement Path MTU Discovery or fragment the inner packet to the correct size.

In many cases it may not be possible or desirable for the tunnel endpoint to interact with the payload, such as when implementing a completely transparent L2 bridge. In these situations, fragmentation of the transit IP header MAY be performed to ensure connectivity. If a packet is fragmented endpoints SHOULD use the path MTU of the transit link to ensure a size is chosen such that fragmentation is only required once between endpoints. Note that some implementations may not be capable of supporting fragmentation or other less common features of the IP header, such as options and extension headers.

#### 4.1.2. DSCP and ECN

When encapsulating IP (including over Ethernet) frames in Geneve, there are several options for propagating DSCP and ECN bits from the inner header to the tunnel on transmission and the reverse on reception.

[RFC2983] lists considerations for mapping DSCP between inner and outer IP headers. Network virtualization is typically more closely aligned with the Pipe model described, where the DSCP value on the tunnel header is set based on a policy (which may be a fixed value, one based on the inner traffic class, or some other mechanism for grouping traffic). Aspects of the Uniform model (which treats the inner and outer DSCP value as a single field by copying on ingress and egress) may also apply, such as the ability to remark the inner header on tunnel egress based on transit marking. However, the Uniform model is not conceptually consistent with network virtualization, which seeks to provide strong isolation between encapsulated traffic and the physical network.

[RFC6040] describes the mechanism for exposing ECN capabilities on IP tunnels and propagating congestion markers to the inner packets. This behavior SHOULD be followed for IP packets encapsulated in Geneve.

#### 4.1.3. Broadcast and Multicast

Geneve tunnels may either be point-to-point unicast between two endpoints or may utilize broadcast or multicast addressing. It is not required that inner and outer addressing match in this respect. For example, in physical networks that do not support multicast,

encapsulated multicast traffic may be replicated into multiple unicast tunnels or forwarded by policy to a unicast location (possibly to be replicated there).

With physical networks that do support multicast it may be desirable to use this capability to take advantage of hardware replication for encapsulated packets. In this case, multicast addresses may be allocated in the physical network corresponding to tenants, encapsulated multicast groups, or some other factor. The allocation of these groups is a component of the control plane and therefore outside of the scope of this document. When physical multicast is in use, the 'C' bit in the Geneve header may be used with groups of devices with heterogeneous capabilities as each device can interpret only the options that are significant to it if they are not critical.

#### 4.2. NIC Offloads

Modern NICs currently provide a variety of offloads to enable the efficient processing of packets. The implementation of many of these offloads requires only that the encapsulated packet be easily parsed (for example, checksum offload). However, optimizations such as LSO and LRO involve some processing of the options themselves since they must be replicated/merged across multiple packets. In these situations, it is desirable to not require changes to the offload logic to handle the introduction of new options. To enable this, some constraints are placed on the definitions of options to allow for simple processing rules:

- o When performing LSO, a NIC MUST replicate the entire Geneve header and all options, including those unknown to the device, onto each resulting segment. However, a given option definition may override this rule and specify different behavior in supporting devices. Conversely, when performing LRO, a NIC MAY assume that a binary comparison of the options (including unknown options) is sufficient to ensure equality and MAY merge packets with equal Geneve headers.
- o Option ordering is not significant and packets with the same options in a different order MAY be processed alike.
- o NICs performing offloads MUST NOT drop packets with unknown options, including those marked as critical.

There is no requirement that a given implementation of Geneve employ the offloads listed as examples above. However, as these offloads are currently widely deployed in commercially available NICs, the rules described here are intended to enable efficient handling of current and future options across a variety of devices.

#### 4.3. Inner VLAN Handling

Geneve is capable of encapsulating a wide range of protocols and therefore a given implementation is likely to support only a small subset of the possibilities. However, as Ethernet is expected to be widely deployed, it is useful to describe the behavior of VLANs inside encapsulated Ethernet frames.

As with any protocol, support for inner VLAN headers is OPTIONAL. In many cases, the use of encapsulated VLANs may be disallowed due to security or implementation considerations. However, in other cases trunking of VLAN frames across a Geneve tunnel can prove useful. As a result, the processing of inner VLAN tags upon ingress or egress from a tunnel endpoint is based upon the configuration of the endpoint and/or control plane and not explicitly defined as part of the data format.

#### 5. Interoperability Issues

Viewed exclusively from the data plane, Geneve does not introduce any interoperability issues as it appears to most devices as UDP frames. However, as there are already a number of tunnel protocols deployed in network virtualization environments, there is a practical question of transition and coexistence.

Since Geneve is a superset of the functionality of the three most common protocols used for network virtualization (VXLAN, NVGRE, and STT) it should be straightforward to port an existing control plane to run on top of it with minimal effort. With both the old and new frame formats supporting the same set of capabilities, there is no need for a hard transition - endpoints directly communicating with each other use any common protocol, which may be different even within a single overall system. As transit devices are primarily forwarding frames on the basis of the IP header, all protocols appear similar and these devices do not introduce additional interoperability concerns.

In order to assist with this transition, it is strongly suggested that implementations support simultaneous operation of both Geneve and existing tunnel protocols as it is expected to be common for a single node to communicate with a mixture of other nodes. Eventually, older protocols may be phased out as they are no longer in use.

## 6. Security Considerations

As UDP/IP packets, Geneve does not have any inherent security mechanisms. As a result, an attacker with access to the underlay network transporting the IP frames has the ability to snoop or inject packets. Legitimate but malicious tunnel endpoints may also spoof identifiers in the tunnel header to gain access to networks owned by other tenants.

Within a particular security domain, such as a data center operated by a single provider, the most common and highest performing security mechanism is isolation of trusted components. Tunnel traffic can be carried over a separate VLAN and filtered at any untrusted boundaries. In addition, tunnel endpoints should only be operated in environments controlled by the service provider, such as the hypervisor itself rather than within a customer VM.

When crossing an untrusted link, such as the public Internet, IPsec [RFC4301] may be used to provide authentication and/or encryption of the IP packets. If the remote tunnel endpoint is not completely trusted, for example it resides on a customer premises, then it may also be necessary to sanitize any tunnel metadata to prevent tenant-hopping attacks.

## 7. IANA Considerations

A UDP destination port in the user range (1024-49151) will be requested from IANA.

In addition, IANA will be requested to create a "Geneve Option Class" registry to allocate Option Classes. This shall be a registry of 16-bit hexadecimal values along with descriptive strings. The identifiers 0x0-0xFF are to be reserved for standardized options for allocation by an IETF working group document or RFC and 0xFFFF for experimental use. Otherwise, identifiers are to be assigned to any organization with an interest in creating Geneve options on a first come first serve basis.

## 8. Acknowledgements

The authors wish to thank Martin Casado, Bruce Davie and Dave Thaler for their input, feedback, and helpful suggestions.

## 9. References

## 9.1. Normative References

- [RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

## 9.2. Informative References

- [ETYPES] The IEEE Registration Authority, "IEEE 802 Numbers", 2013, <<http://www.iana.org/assignments/ieee-802-numbers/ieee-802-numbers.xml>>.
- [I-D.davie-stt] Davie, B. and J. Gross, "A Stateless Transport Tunneling Protocol for Network Virtualization (STT)", draft-davie-stt-04 (work in progress), September 2013.
- [I-D.ietf-nvo3-dataplane-requirements] Bitar, N., Lasserre, M., Balus, F., Morin, T., Jin, L., and B. Khasnabish, "NVO3 Data Plane Requirements", draft-ietf-nvo3-dataplane-requirements-02 (work in progress), November 2013.
- [I-D.mahalingam-dutt-dcops-vxlan] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", draft-mahalingam-dutt-dcops-vxlan-08 (work in progress), February 2014.
- [I-D.sridharan-virtualization-nvgre] Sridharan, M., Greenberg, A., Wang, Y., Garg, P., Venkataramiah, N., Duda, K., Ganga, I., Lin, G., Pearson, M., Thaler, P., and C. Tumuluri, "NVGRE: Network Virtualization using Generic Routing Encapsulation", draft-sridharan-virtualization-nvgre-04 (work in progress), February 2014.
- [IEEE.802.1Q-2011] IEEE, "IEEE Standard for Local and metropolitan area networks -- Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks", IEEE Std 802.1Q, 2011.
- [RFC2983] Black, D., "Differentiated Services and Tunnels", RFC 2983, October 2000.

- [RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", RFC 6040, November 2010.
- [RFC6935] Eubanks, M., Chimento, P., and M. Westerlund, "IPv6 and UDP Checksums for Tunneled Packets", RFC 6935, April 2013.
- [VL2] Greenberg et al, , "VL2: A Scalable and Flexible Data Center Network", 2009.
- Proc. ACM SIGCOMM 2009

## Authors' Addresses

Jesse Gross  
VMware, Inc.  
3401 Hillview Ave.  
Palo Alto, CA 94304  
USA  
  
Email: jgross@vmware.com

T. Sridhar  
VMware, Inc.  
3401 Hillview Ave.  
Palo Alto, CA 94304  
USA  
  
Email: tsridhar@vmware.com

Pankaj Garg  
Microsoft Corporation  
1 Microsoft Way  
Redmond, WA 98052  
USA  
  
Email: pankajg@microsoft.com



Chris Wright  
Red Hat Inc.  
1801 Varsity Drive  
Raleigh, NC 27606  
USA

Email: [chrisw@redhat.com](mailto:chrisw@redhat.com)

Ilango Ganga  
Intel Corporation  
2200 Mission College Blvd.  
Santa Clara, CA 95054  
USA

Email: [ilango.s.ganga@intel.com](mailto:ilango.s.ganga@intel.com)

Internet Engineering Task Force  
Internet-Draft  
Intended status: Informational  
Expires: August 18, 2014

D. Black  
EMC  
J. Hudson  
Brocade  
L. Kreeger  
Cisco  
M. Lasserre  
Alcatel-Lucent  
T. Narten  
IBM  
February 14, 2014

An Architecture for Overlay Networks (NVO3)  
draft-ietf-nvo3-arch-01

Abstract

This document presents a high-level overview architecture for building overlay networks in NVO3. The architecture is given at a high-level, showing the major components of an overall system. An important goal is to divide the space into individual smaller components that can be implemented independently and with clear interfaces and interactions with other components. It should be possible to build and implement individual components in isolation and have them work with other components with no changes to other components. That way implementers have flexibility in implementing individual components and can optimize and innovate within their respective components without requiring changes to other components.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 18, 2014.

## Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Terminology . . . . .	3
3. Background . . . . .	4
3.1. VN Service (L2 and L3) . . . . .	6
3.1.1. VLAN Tags in L2 Service . . . . .	7
3.1.2. TTL Considerations . . . . .	7
3.2. Network Virtualization Edge (NVE) . . . . .	7
3.3. Network Virtualization Authority (NVA) . . . . .	9
3.4. VM Orchestration Systems . . . . .	9
4. Network Virtualization Edge (NVE) . . . . .	10
4.1. NVE Co-located With Server Hypervisor . . . . .	11
4.2. Split-NVE . . . . .	11
4.2.1. Tenant VLAN handling in Split-NVE Case . . . . .	12
4.3. NVE State . . . . .	12
4.4. Multi-Homing of NVEs . . . . .	13
4.5. VAP . . . . .	14
5. Tenant System Types . . . . .	14
5.1. Overlay-Aware Network Service Appliances . . . . .	14
5.2. Bare Metal Servers . . . . .	15
5.3. Gateways . . . . .	15
5.4. Distributed Inter-VN Gateways . . . . .	16
5.5. ARP and Neighbor Discovery . . . . .	17
6. NVE-NVE Interaction . . . . .	17
7. Network Virtualization Authority . . . . .	18
7.1. How an NVA Obtains Information . . . . .	18
7.2. Internal NVA Architecture . . . . .	19
7.3. NVA External Interface . . . . .	19
8. NVE-to-NVA Protocol . . . . .	21
8.1. NVE-NVA Interaction Models . . . . .	21
8.2. Direct NVE-NVA Protocol . . . . .	22
8.3. Propagating Information Between NVEs and NVAs . . . . .	23

9. Federated NVAs . . . . .	24
9.1. Inter-NVA Peering . . . . .	26
10. Control Protocol Work Areas . . . . .	26
11. NVO3 Data Plane Encapsulation . . . . .	27
12. Operations and Management . . . . .	28
13. Summary . . . . .	28
14. Acknowledgments . . . . .	28
15. IANA Considerations . . . . .	28
16. Security Considerations . . . . .	28
17. Informative References . . . . .	28
Appendix A. Change Log . . . . .	30
A.1. Changes From draft-ietf-nvo3-arch-00 to -01 . . . . .	30
A.2. Changes From draft-narten-nvo3 to draft-ietf-nvo3 . . . . .	31
A.3. Changes From -00 to -01 (of draft-narten-nvo3-arch) . . . . .	31
Authors' Addresses . . . . .	31

## 1. Introduction

This document presents a high-level architecture for building overlay networks in NVO3. The architecture is given at a high-level, showing the major components of an overall system. An important goal is to divide the space into smaller individual components that can be implemented independently and with clear interfaces and interactions with other components. It should be possible to build and implement individual components in isolation and have them work with other components with no changes to other components. That way implementers have flexibility in implementing individual components and can optimize and innovate within their respective components without necessarily requiring changes to other components.

The motivation for overlay networks is given in [I-D.ietf-nvo3-overlay-problem-statement]. "Framework for DC Network Virtualization" [I-D.ietf-nvo3-framework] provides a framework for discussing overlay networks generally and the various components that must work together in building such systems. This document differs from the framework document in that it doesn't attempt to cover all possible approaches within the general design space. Rather, it describes one particular approach.

This document is intended to be a concrete strawman that can be used for discussion within the IETF NVO3 WG on what the NVO3 architecture should look like.

## 2. Terminology

This document uses the same terminology as [I-D.ietf-nvo3-framework]. In addition, the following terms are used:

**NV Domain** A Network Virtualization Domain is an administrative construct that defines a Network Virtualization Authority (NVA), the set of Network Virtualization Edges (NVEs) associated with that NVA, and the set of virtual networks the NVA manages and supports. NVEs are associated with a (logically centralized) NVA, and an NVE supports communication for any of the virtual networks in the domain.

**NV Region** A region over which information about a set of virtual networks is shared. The degenerate case of a single NV Domain corresponds to an NV region corresponding to that domain. The more interesting case occurs when two or more NV Domains share information about part or all of a set of virtual networks that they manage. Two NVAs share information about particular virtual networks for the purpose of supporting connectivity between tenants located in different NVA Domains. NVAs can share information about an entire NV domain, or just individual virtual networks.

**Tenant System Identifier (TSI)** Interface to a Virtual Network as presented to a Tenant System. The TSI logically connects to the NVE via a Virtual Access Point (VAP). To the Tenant System, the TSI is like a NIC; the TSI presents itself to a Tenant System as a normal network interface.

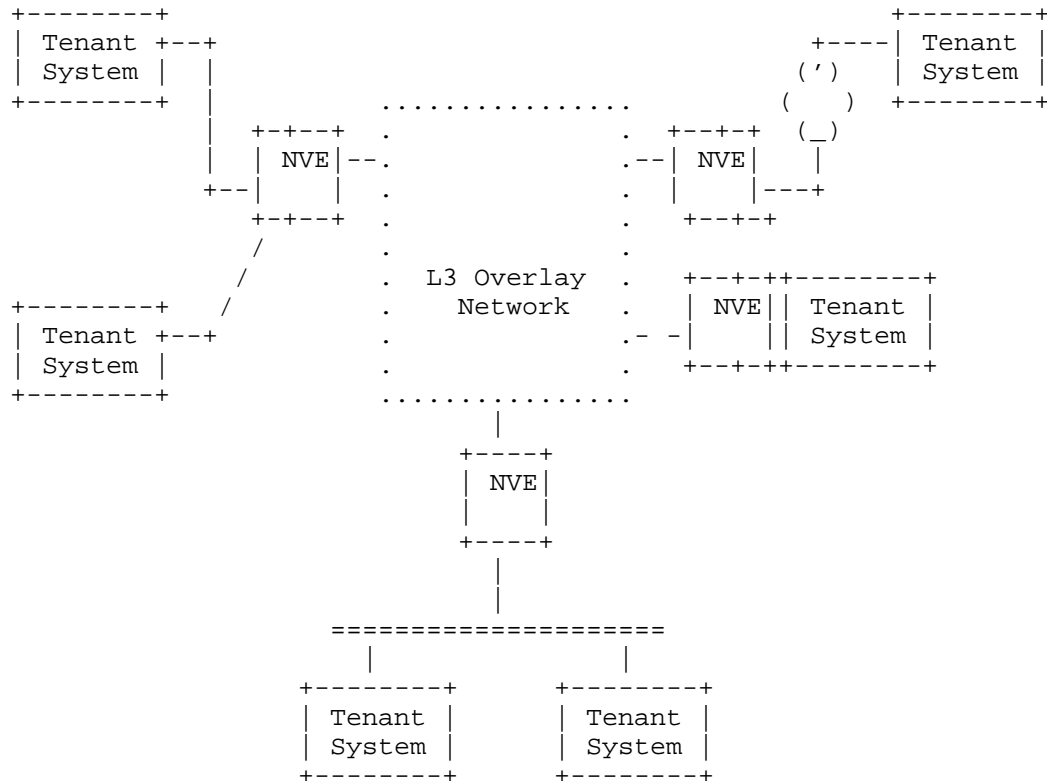
**VLAN** Unless stated otherwise, the terms VLAN and VLAN Tag are used in this document denote a C-VLAN [IEEE-802.1Q] and the terms are used interchangeably to improve readability.

### 3. Background

Overlay networks are an approach for providing network virtualization services to a set of Tenant Systems (TSs) [I-D.ietf-nvo3-framework]. With overlays, data traffic between tenants is tunneled across the underlying data center's IP network. The use of tunnels provides a number of benefits by decoupling the network as viewed by tenants from the underlying physical network across which they communicate.

Tenant Systems connect to Virtual Networks (VNs), with each VN having associated attributes defining properties of the network, such as the set of members that connect to it. Tenant Systems connected to a virtual network typically communicate freely with other Tenant Systems on the same VN, but communication between Tenant Systems on one VN and those external to the VN (whether on another VN or connected to the Internet) is carefully controlled and governed by policy.

A Network Virtualization Edge (NVE) [I-D.ietf-nvo3-framework] is the entity that implements the overlay functionality. An NVE resides at the boundary between a Tenant System and the overlay network as shown in Figure 1. An NVE creates and maintains local state about each Virtual Network for which it is providing service on behalf of a Tenant System.



The dotted line indicates a network connection (i.e., IP).

Figure 1: NVO3 Generic Reference Model

The following subsections describe key aspects of an overlay system in more detail. Section 3.1 describes the service model (Ethernet vs. IP) provided to Tenant Systems. Section 3.2 describes NVEs in more detail. Section 3.3 introduces the Network Virtualization Authority, from which NVEs obtain information about virtual networks. Section 3.4 provides background on VM orchestration systems and their use of virtual networks.

### 3.1. VN Service (L2 and L3)

A Virtual Network provides either L2 or L3 service to connected tenants. For L2 service, VNs transport Ethernet frames, and a Tenant System is provided with a service that is analogous to being connected to a specific L2 C-VLAN. L2 broadcast frames are generally delivered to all (and multicast frames delivered to a subset of) the other Tenant Systems on the VN. To a Tenant System, it appears as if they are connected to a regular L2 Ethernet link. Within NVO3, tenant frames are tunneled to remote NVEs based on the MAC addresses of the frame headers as originated by the Tenant System. On the underlay, NVO3 packets are forwarded between NVEs based on the outer addresses of tunneled packets.

For L3 service, VNs transport IP datagrams, and a Tenant System is provided with a service that only supports IP traffic. Within NVO3, tenant frames are tunneled to remote NVEs based on the IP addresses of the packet originated by the Tenant System; any L2 destination addresses provided by Tenant Systems are effectively ignored.

L2 service is intended for systems that need native L2 Ethernet service and the ability to run protocols directly over Ethernet (i.e., not based on IP). L3 service is intended for systems in which all the traffic can safely be assumed to be IP. It is important to note that whether NVO3 provides L2 or L3 service to a Tenant System, the Tenant System does not generally need to be aware of the distinction. In both cases, the virtual network presents itself to the Tenant System as an L2 Ethernet interface. An Ethernet interface is used in both cases simply as a widely supported interface type that essentially all Tenant Systems already support. Consequently, no special software is needed on Tenant Systems to use an L3 vs. an L2 overlay service.

NVO3 can also provide a combined L2 and L3 service to tenants. A combined service provides L2 service for intra-VN communication, but also provides L3 service for L3 traffic entering or leaving the VN. Architecturally, the handling of a combined L2/L3 service in NVO3 is intended to match what is commonly done today in non-overlay environments by devices providing a combined bridge/router service. With combined service, the virtual network itself retains the semantics of L2 service and all traffic is processed according to its L2 semantics. In addition, however, traffic requiring IP processing is also processed at the IP level.

The IP processing for a combined service can be implemented on a standalone device attached to the virtual network (e.g., an IP router) or implemented locally on the NVE (see Section 5.4 on Distributed Gateways). For unicast traffic, NVE implementation of a

combined service may result in a packet being delivered to another TS attached to the same NVE (on either the same or a different VN) or tunneled to a remote NVE, or even forwarded outside the NVO3 domain. For multicast or broadcast packets, the combination of NVE L2 and L3 processing may result in copies of the packet receiving both L2 and L3 treatments to realize delivery to all of the destinations involved. This optimized NVE implementation of IP routing results in the same network delivery behavior as if the L2 processing of the packet included delivery of the packet to an IP router attached to the L2 VN as a TS, with the router having additional network attachments to other networks, either virtual or not.

#### 3.1.1. VLAN Tags in L2 Service

An NVO3 L2 virtual network service may include encapsulated L2 VLAN tags provided by a Tenant System, but does not use encapsulated tags in deciding where and how to forward traffic. Such VLAN tags can be passed through, so that Tenant Systems that send or expect to receive them can be supported as appropriate.

The processing of VLAN tags that an NVE receives from a TS is controlled by settings associated with the VAP. Just as in the case with ports on Ethernet switches, a number of settings could be imagined. For example, C-TAGs can be passed through transparently, they could always be stripped upon receipt from a Tenant System, they could be compared against a list of explicitly configured tags, etc.

Note that the handling of C-VIDs has additional complications, as described in Section 4.2.1 below.

#### 3.1.2. TTL Considerations

For L3 service, Tenant Systems should expect the TTL of the packets they send to be decremented by at least 1. For L2 service, the TTL on packets (when the packet is IP) is not modified.

### 3.2. Network Virtualization Edge (NVE)

Tenant Systems connect to NVEs via a Tenant System Interface (TSI). The TSI logically connects to the NVE via a Virtual Access Point (VAP) and each VAP is associated with one Virtual Network as shown in Figure 2. To the Tenant System, the TSI is like a NIC; the TSI presents itself to a Tenant System as a normal network interface. On the NVE side, a VAP is a logical network port (virtual or physical) into a specific virtual network. Note that two different Tenant Systems (and TSIs) attached to a common NVE can share a VAP (e.g., TS1 and TS2 in Figure 2) so long as they connect to the same Virtual Network.



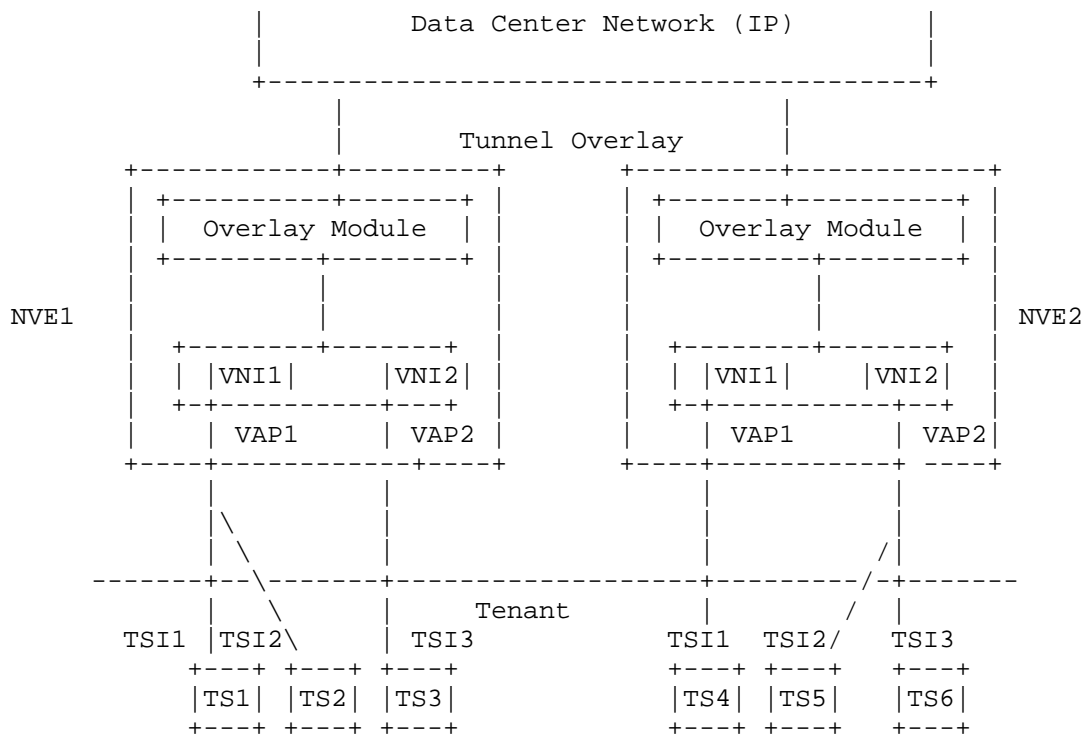


Figure 2: NVE Reference Model

The Overlay Module performs the actual encapsulation and decapsulation of tunneled packets. The NVE maintains state about the virtual networks it is a part of so that it can provide the Overlay Module with such information as the destination address of the NVE to tunnel a packet to, or the Context ID that should be placed in the encapsulation header to identify the virtual network that a tunneled packet belongs to.

On the data center network side, the NVE sends and receives native IP traffic. When ingressing traffic from a Tenant System, the NVE identifies the egress NVE to which the packet should be sent, adds an overlay encapsulation header, and sends the packet on the underlay network. When receiving traffic from a remote NVE, an NVE strips off the encapsulation header, and delivers the (original) packet to the appropriate Tenant System.

Conceptually, the NVE is a single entity implementing the NVO3 functionality. In practice, there are a number of different implementation scenarios, as described in detail in Section 4.

### 3.3. Network Virtualization Authority (NVA)

Address dissemination refers to the process of learning, building and distributing the mapping/forwarding information that NVEs need in order to tunnel traffic to each other on behalf of communicating Tenant Systems. For example, in order to send traffic to a remote Tenant System, the sending NVE must know the destination NVE for that Tenant System.

One way to build and maintain mapping tables is to use learning, as 802.1 bridges do [IEEE-802.1Q]. When forwarding traffic to multicast or unknown unicast destinations, an NVE could simply flood traffic everywhere. While flooding works, it can lead to traffic hot spots and can lead to problems in larger networks.

Alternatively, to avoid issues associated with flooding, NVEs can make use of a Network Virtualization Authority (NVA). An NVA is the entity that provides address mapping and other information to NVEs. NVEs interact with an NVA to obtain any required address mapping information they need in order to properly forward traffic on behalf of tenants. The term NVA refers to the overall system, without regards to its scope or how it is implemented. NVAs provide a service, and NVEs access that service via an NVE-to-NVA protocol as discussed in Section 4.3.

Even when an NVA is present, Ethernet bridge MAC address learning could be used as a fallback mechanism, should the NVA be unable to provide an answer or for other reasons. This document does not consider flooding approaches in detail, as there are a number of benefits in using an approach that depends on the presence of an NVA.

For the rest of this document, it is assumed that an NVA exists and will be used. NVAs are discussed in more detail in Section 7.

### 3.4. VM Orchestration Systems

VM Orchestration systems manage server virtualization across a set of servers. Although VM management is a separate topic from network virtualization, the two areas are closely related. Managing the creation, placement, and movement of VMs also involves creating, attaching to and detaching from virtual networks. A number of existing VM orchestration systems have incorporated aspects of virtual network management into their systems.

When a new VM image is started, the VM Orchestration system determines where the VM should be placed, interacts with the hypervisor on the target server to load and start the VM and controls when a VM should be shutdown or migrated elsewhere. VM Orchestration

systems also have knowledge about how a VM should connect to a network, possibly including the name of the virtual network to which a VM is to connect. The VM orchestration system can pass such information to the hypervisor when a VM is instantiated. VM orchestration systems have significant (and sometimes global) knowledge over the domain they manage. They typically know on what servers a VM is running, and meta data associated with VM images can be useful from a network virtualization perspective. For example, the meta data may include the addresses (MAC and IP) the VMs will use and the name(s) of the virtual network(s) they connect to.

VM orchestration systems run a protocol with an agent running on the hypervisor of the servers they manage. That protocol can also carry information about what virtual network a VM is associated with. When the orchestrator instantiates a VM on a hypervisor, the hypervisor interacts with the NVE in order to attach the VM to the virtual networks it has access to. In general, the hypervisor will need to communicate significant VM state changes to the NVE. In the reverse direction, the NVE may need to communicate network connectivity information back to the hypervisor. Example VM orchestration systems in use today include VMware's vCenter Server or Microsoft's System Center Virtual Machine Manager. Both can pass information about what virtual networks a VM connects to down to the hypervisor. The protocol used between the VM orchestration system and hypervisors is generally proprietary.

It should be noted that VM orchestration systems may not have direct access to all networking related information a VM uses. For example, a VM may make use of additional IP or MAC addresses that the VM management system is not aware of.

#### 4. Network Virtualization Edge (NVE)

As introduced in Section 3.2 an NVE is the entity that implements the overlay functionality. This section describes NVEs in more detail. An NVE will have two external interfaces:

**Tenant System Facing:** On the Tenant System facing side, an NVE interacts with the hypervisor (or equivalent entity) to provide the NVO3 service. An NVE will need to be notified when a Tenant System "attaches" to a virtual network (so it can validate the request and set up any state needed to send and receive traffic on behalf of the Tenant System on that VN). Likewise, an NVE will need to be informed when the Tenant System "detaches" from the virtual network so that it can reclaim state and resources appropriately.

Data Center Network (DCN) Facing: On the data center network facing side, an NVE interfaces with the data center underlay network, sending and receiving tunneled TS packets to and from the underlay. The NVE may also run a control protocol with other entities on the network, such as the Network Virtualization Authority.

#### 4.1. NVE Co-located With Server Hypervisor

When server virtualization is used, the entire NVE functionality will typically be implemented as part of the hypervisor and/or virtual switch on the server. In such cases, the Tenant System interacts with the hypervisor and the hypervisor interacts with the NVE. Because the interaction between the hypervisor and NVE is implemented entirely in software on the server, there is no "on-the-wire" protocol between Tenant Systems (or the hypervisor) and the NVE that needs to be standardized. While there may be APIs between the NVE and hypervisor to support necessary interaction, the details of such an API are not in-scope for the IETF to work on.

Implementing NVE functionality entirely on a server has the disadvantage that server CPU resources must be spent implementing the NVO3 functionality. Experimentation with overlay approaches and previous experience with TCP and checksum adapter offloads suggests that offloading certain NVE operations (e.g., encapsulation and decapsulation operations) onto the physical network adapter can produce performance improvements. As has been done with checksum and /or TCP server offload and other optimization approaches, there may be benefits to offloading common operations onto adapters where possible. Just as important, the addition of an overlay header can disable existing adapter offload capabilities that are generally not prepared to handle the addition of a new header or other operations associated with an NVE.

While the details of how to split the implementation of specific NVE functionality between a server and its network adapters is outside the scope of IETF standardization, the NVO3 architecture should support such separation. Ideally, it may even be possible to bypass the hypervisor completely on critical data path operations so that packets between a TS and its VN can be sent and received without having the hypervisor involved in each individual packet operation.

#### 4.2. Split-NVE

Another possible scenario leads to the need for a split NVE implementation. An NVE running on a server (e.g. within a hypervisor) could support NVO3 towards the tenant, but not perform all NVE functions (e.g., encapsulation) directly on the server; some

of the actual NVO3 functionality could be implemented on (i.e., offloaded to) an adjacent switch to which the server is attached. While one could imagine a number of link types between a server and the NVE, one simple deployment scenario would involve a server and NVE separated by a simple L2 Ethernet link. A more complicated scenario would have the server and NVE separated by a bridged access network, such as when the NVE resides on a ToR, with an embedded switch residing between servers and the ToR.

While the above talks about a scenario involving a hypervisor, it should be noted that the same scenario can apply to Network Service Appliances as discussed in Section 5.1. In general, when this document discusses the interaction between a hypervisor and NVE, the discussion applies to Network Service Appliances as well.

For the split NVE case, protocols will be needed that allow the hypervisor and NVE to negotiate and setup the necessary state so that traffic sent across the access link between a server and the NVE can be associated with the correct virtual network instance. Specifically, on the access link, traffic belonging to a specific Tenant System would be tagged with a specific VLAN C-TAG that identifies which specific NVO3 virtual network instance it belongs to. The hypervisor-NVE protocol would negotiate which VLAN C-TAG to use for a particular virtual network instance. More details of the protocol requirements for functionality between hypervisors and NVEs can be found in [I-D.kreeger-nvo3-hypervisor-nve-cp].

#### 4.2.1. Tenant VLAN handling in Split-NVE Case

Preserving tenant VLAN tags across a NVO3 as described in Section 3.1.1 poses additional complications in the split-NVE case. The portion of the NVE that performs the encapsulation function needs access to the specific VLAN tags that the Tenant System is using in order to include them in the encapsulated packet. When an NVE is implemented entirely within the hypervisor, the NVE has access to the complete original packet (including any VLAN tags) sent by the tenant. In the split-NVE case, however, the VLAN tag used between the hypervisor and offloaded portions of the NVE normally only identify the specific VN that traffic belongs. In order to allow a tenant to preserve VLAN information in the split-NVE case, additional mechanisms would be needed.

#### 4.3. NVE State

NVEs maintain internal data structures and state to support the sending and receiving of tenant traffic. An NVE may need some or all of the following information:

1. An NVE keeps track of which attached Tenant Systems are connected to which virtual networks. When a Tenant System attaches to a virtual network, the NVE will need to create or update local state for that virtual network. When the last Tenant System detaches from a given VN, the NVE can reclaim state associated with that VN.
2. For tenant unicast traffic, an NVE maintains a per-VN table of mappings from Tenant System (inner) addresses to remote NVE (outer) addresses.
3. For tenant multicast (or broadcast) traffic, an NVE maintains a per-VN table of mappings and other information on how to deliver multicast (or broadcast) traffic. If the underlying network supports IP multicast, the NVE could use IP multicast to deliver tenant traffic. In such a case, the NVE would need to know what IP underlay multicast address to use for a given VN. Alternatively, if the underlying network does not support multicast, an NVE could use serial unicast to deliver traffic. In such a case, an NVE would need to know which destinations are subscribers to the tenant multicast group. An NVE could use both approaches, switching from one mode to the other depending on such factors as bandwidth efficiency and group membership sparseness.
4. An NVE maintains necessary information to encapsulate outgoing traffic, including what type of encapsulation and what value to use for a Context ID within the encapsulation header.
5. In order to deliver incoming encapsulated packets to the correct Tenant Systems, an NVE maintains the necessary information to map incoming traffic to the appropriate VAP and Tenant System.
6. An NVE may find it convenient to maintain additional per-VN information such as QoS settings, Path MTU information, ACLs, etc.

#### 4.4. Multi-Homing of NVEs

NVEs may be multi-homed. That is, an NVE may have more than one IP address associated with it on the underlay network. Multihoming happens in two different scenarios. First, an NVE may have multiple interfaces connecting it to the underlay. Each of those interfaces will typically have a different IP address, resulting in a specific Tenant Address (on a specific VN) being reachable through the same NVE but through more than one underlay IP address. Second, a specific tenant system may be reachable through more than one NVE, each having one or more underlay addresses. In both cases, the NVE

address mapping tables need to support one-to-many mappings and enable a sending NVE to (at a minimum) be able to fail over from one IP address to another, e.g., should a specific NVE underlay address become unreachable.

Multi-homing is needed to support important use cases. First, a bare metal server may have multiple uplink connections to either the same or different NVEs. Having only a single physical path to an upstream NVE, or indeed, having all traffic flow through a single NVE would be considered unacceptable in highly-resilient deployment scenarios that seek to avoid single points of failure. Moreover, in today's networks, the availability of multiple paths would require that they be usable in an active-active fashion (e.g., for load balancing).

#### 4.5. VAP

The VAP is the NVE-side of the interface between the NVE and the TS. Traffic to and from the tenant flows through the VAP. If an NVE runs into difficulties sending traffic received on the VAP, it may need signal such errors back to the VAP. Because the VAP is an emulation of a physical port, its ability to signal NVE errors is limited and lacks sufficient granularity to reflect all possible errors an NVE may encounter (e.g., inability reach a particular destination). For some errors, such as an NVE losing all of its connections to the underlay, could be reflected back to the VAP by effectively disabling it. This state change would reflect itself on the TS as an interface going down, allowing the TS to implement interface error handling, e.g., failover, in the same manner as when a physical interfaces becomes disabled.

### 5. Tenant System Types

This section describes a number of special Tenant System types and how they fit into an NVO3 system.

#### 5.1. Overlay-Aware Network Service Appliances

Some Network Service Appliances [I-D.ietf-nvo3-nve-nva-cp-req] (virtual or physical) provide tenant-aware services. That is, the specific service they provide depends on the identity of the tenant making use of the service. For example, firewalls are now becoming available that support multi-tenancy where a single firewall provides virtual firewall service on a per-tenant basis, using per-tenant configuration rules and maintaining per-tenant state. Such appliances will be aware of the VN an activity corresponds to while processing requests. Unlike server virtualization, which shields VMs from needing to know about multi-tenancy, a Network Service Appliance may explicitly support multi-tenancy. In such cases, the Network

Service Appliance itself will be aware of network virtualization and either embed an NVE directly, or implement a split NVE as described in Section 4.2. Unlike server virtualization, however, the Network Service Appliance may not be running a traditional hypervisor and the VM Orchestration system may not interact with the Network Service Appliance. The NVE on such appliances will need to support a control plane to obtain the necessary information needed to fully participate in an NVO3 Domain.

## 5.2. Bare Metal Servers

Many data centers will continue to have at least some servers operating as non-virtualized (or "bare metal") machines running a traditional operating system and workload. In such systems, there will be no NVE functionality on the server, and the server will have no knowledge of NVO3 (including whether overlays are even in use). In such environments, the NVE functionality can reside on the first-hop physical switch. In such a case, the network administrator would (manually) configure the switch to enable the appropriate NVO3 functionality on the switch port connecting the server and associate that port with a specific virtual network. Such configuration would typically be static, since the server is not virtualized, and once configured, is unlikely to change frequently. Consequently, this scenario does not require any protocol or standards work.

## 5.3. Gateways

Gateways on VNs relay traffic onto and off of a virtual network. Tenant Systems use gateways to reach destinations outside of the local VN. Gateways receive encapsulated traffic from one VN, remove the encapsulation header, and send the native packet out onto the data center network for delivery. Outside traffic enters a VN in a reverse manner.

Gateways can be either virtual (i.e., implemented as a VM) or physical (i.e., as a standalone physical device). For performance reasons, standalone hardware gateways may be desirable in some cases. Such gateways could consist of a simple switch forwarding traffic from a VN onto the local data center network, or could embed router functionality. On such gateways, network interfaces connecting to virtual networks will (at least conceptually) embed NVE (or split-NVE) functionality within them. As in the case with Network Service Appliances, gateways may not support a hypervisor and will need an appropriate control plane protocol to obtain the information needed to provide NVO3 service.

Gateways handle several different use cases. For example, a virtual network could consist of systems supporting overlays together with



legacy Tenant Systems that do not. Gateways could be used to connect legacy systems supporting, e.g., L2 VLANs, to specific virtual networks, effectively making them part of the same virtual network. Gateways could also forward traffic between a virtual network and other hosts on the data center network or relay traffic between different VNs. Finally, gateways can provide external connectivity such as Internet or VPN access.

#### 5.4. Distributed Inter-VN Gateways

The relaying of traffic from one VN to another deserves special consideration. Whether traffic is permitted to flow from one VN to another is a matter of policy, and would not (by default) be allowed unless explicitly enabled. In addition, NVAs are the logical place to maintain policy information about allowed inter-VN communication. Policy enforcement for inter-VN communication can be handled in (at least) two different ways. Explicit gateways could be the central point for such enforcement, with all inter-VN traffic forwarded to such gateways for processing. Alternatively, the NVA can provide such information directly to NVEs, by either providing a mapping for a target TS on another VN, or indicating that such communication is disallowed by policy.

When inter-VN gateways are centralized, traffic between TSs on different VNs can take suboptimal paths, i.e., triangular routing results in paths that always traverse the gateway. In the worst case, traffic between two TSs connected to the same NVE can be hair-pinned through an external gateway. As an optimization, individual NVEs can be part of a distributed gateway that performs such relaying, reducing or completely eliminating triangular routing. In a distributed gateway, each ingress NVE can perform such relaying activity directly, so long as it has access to the policy information needed to determine whether cross-VN communication is allowed. Having individual NVEs be part of a distributed gateway allows them to tunnel traffic directly to the destination NVE without the need to take suboptimal paths.

The NVO3 architecture must support distributed gateways for the case of inter-VN communication. Such support requires that NVO3 control protocols include mechanisms for the maintenance and distribution of policy information about what type of cross-VN communication is allowed so that NVEs acting as distributed gateways can tunnel traffic from one VN to another as appropriate.

Distributed gateways could also be used to distribute other traditional router services to individual NVEs. The NVO3 architecture does not preclude such implementations, but does not define or require them as they are outside the scope of NVO3.

### 5.5. ARP and Neighbor Discovery

For an L2 service, strictly speaking, special processing of ARP [RFC0826] (and IPv6 Neighbor Discovery (ND) [RFC4861]) is not required. ARP requests are broadcast, and NVO3 can deliver ARP requests to all members of a given L2 virtual network, just as it does for any packet sent to an L2 broadcast address. Similarly, ND requests are sent via IP multicast, which NVO3 can support by delivering via L2 multicast. However, as a performance optimization, an NVE can intercept ARP (or ND) requests from its attached TSs and respond to them directly using information in its mapping tables. Since an NVE will have mechanisms for determining the NVE address associated with a given TS, the NVE can leverage the same mechanisms to suppress sending ARP and ND requests for a given TS to other members of the VN. The NVO3 architecture must support such a capability.

### 6. NVE-NVE Interaction

Individual NVEs will interact with each other for the purposes of tunneling and delivering traffic to remote TSs. At a minimum, a control protocol may be needed for tunnel setup and maintenance. For example, tunneled traffic may need to be encrypted or integrity protected, in which case it will be necessary to set up appropriate security associations between NVE peers. It may also be desirable to perform tunnel maintenance (e.g., continuity checks) on a tunnel in order to detect when a remote NVE becomes unreachable. Such generic tunnel setup and maintenance functions are not generally NVO3-specific. Hence, NVO3 expects to leverage existing tunnel maintenance protocols rather than defining new ones.

Some NVE-NVE interactions may be specific to NVO3 (and in particular be related to information kept in mapping tables) and agnostic to the specific tunnel type being used. For example, when tunneling traffic for TS-X to a remote NVE, it is possible that TS-X is not presently associated with the remote NVE. Normally, this should not happen, but there could be race conditions where the information an NVE has learned from the NVA is out-of-date relative to actual conditions. In such cases, the remote NVE could return an error or warning indication, allowing the sending NVE to attempt a recovery or otherwise attempt to mitigate the situation.

The NVE-NVE interaction could signal a range of indications, for example:

- o "No such TS here", upon a receipt of a tunneled packet for an unknown TS.

- o "TS-X not here, try the following NVE instead" (i.e., a redirect).
- o Delivered to correct NVE, but could not deliver packet to TS-X (soft error).
- o Delivered to correct NVE, but could not deliver packet to TS-X (hard error).

When an NVE receives information from a remote NVE that conflicts with the information it has in its own mapping tables, it should consult with the NVA to resolve those conflicts. In particular, it should confirm that the information it has is up-to-date, and it might indicate the error to the NVA, so as to nudge the NVA into following up (as appropriate). While it might make sense for an NVE to update its mapping table temporarily in response to an error from a remote NVE, any changes must be handled carefully as doing so can raise security considerations if the received information cannot be authenticated. That said, a sending NVE might still take steps to mitigate a problem, such as applying rate limiting to data traffic towards a particular NVE or TS.

## 7. Network Virtualization Authority

Before sending to and receiving traffic from a virtual network, an NVE must obtain the information needed to build its internal forwarding tables and state as listed in Section 4.3. An NVE can obtain such information from a Network Virtualization Authority.

The Network Virtualization Authority (NVA) is the entity that is expected to provide address mapping and other information to NVEs. NVEs can interact with an NVA to obtain any required information they need in order to properly forward traffic on behalf of tenants. The term NVA refers to the overall system, without regards to its scope or how it is implemented.

### 7.1. How an NVA Obtains Information

There are two primary ways in which an NVA can obtain the address dissemination information it manages. The NVA can obtain information either from the VM orchestration system, or directly from the NVEs themselves.

On virtualized systems, the NVA may be able to obtain the address mapping information associated with VMs from the VM orchestration system itself. If the VM orchestration system contains a master database for all the virtualization information, having the NVA obtain information directly to the orchestration system would be a natural approach. Indeed, the NVA could effectively be co-located

with the VM orchestration system itself. In such systems, the VM orchestration system communicates with the NVE indirectly through the hypervisor.

However, as described in Section 4 not all NVEs are associated with hypervisors. In such cases, NVAs cannot leverage VM orchestration protocols to interact with an NVE and will instead need to peer directly with them. By peering directly with an NVE, NVAs can obtain information about the TSs connected to that NVE and can distribute information to the NVE about the VNs those TSs are associated with. For example, whenever a Tenant System attaches to an NVE, that NVE would notify the NVA that the TS is now associated with that NVE. Likewise when a TS detaches from an NVE, that NVE would inform the NVA. By communicating directly with NVEs, both the NVA and the NVE are able to maintain up-to-date information about all active tenants and the NVEs to which they are attached.

## 7.2. Internal NVA Architecture

For reliability and fault tolerance reasons, an NVA would be implemented in a distributed or replicated manner without single points of failure. How the NVA is implemented, however, is not important to an NVE so long as the NVA provides a consistent and well-defined interface to the NVE. For example, an NVA could be implemented via database techniques whereby a server stores address mapping information in a traditional (possibly replicated) database. Alternatively, an NVA could be implemented in a distributed fashion using an existing (or modified) routing protocol to maintain and distribute mappings. So long as there is a clear interface between the NVE and NVA, how an NVA is architected and implemented is not important to an NVE.

A number of architectural approaches could be used to implement NVAs themselves. NVAs manage address bindings and distribute them to where they need to go. One approach would be to use Border Gateway Protocol (BGP) [RFC4364] (possibly with extensions) and route reflectors. Another approach could use a transaction-based database model with replicated servers. Because the implementation details are local to an NVA, there is no need to pick exactly one solution technology, so long as the external interfaces to the NVEs (and remote NVAs) are sufficiently well defined to achieve interoperability.

## 7.3. NVA External Interface

[note: the following section discusses various options that the WG has not yet expressed an opinion on. Discussion is encouraged. ]

Conceptually, from the perspective of an NVE, an NVA is a single entity. An NVE interacts with the NVA, and it is the NVA's responsibility for ensuring that interactions between the NVE and NVA result in consistent behavior across the NVA and all other NVEs using the same NVA. Because an NVA is built from multiple internal components, an NVA will have to ensure that information flows to all internal NVA components appropriately.

One architectural question is how the NVA presents itself to the NVE. For example, an NVA could be required to provide access via a single IP address. If NVEs only have one IP address to interact with, it would be the responsibility of the NVA to handle NVA component failures, e.g., by using a "floating IP address" that migrates among NVA components to ensure that the NVA can always be reached via the one address. Having all NVA accesses through a single IP address, however, adds constraints to implementing robust failover, load balancing, etc.

[Note: the following is a strawman proposal.]

In the NVO3 architecture, an NVA is accessed through one or more IP addresses (or IP address/port combination). If multiple IP addresses are used, each IP address provides equivalent functionality, meaning that an NVE can use any of the provided addresses to interact with the NVA. Should one address stop working, an NVE is expected to failover to another. While the different addresses result in equivalent functionality, one address may be more respond more quickly than another, e.g., due to network conditions, load on the server, etc.

[Note: should we support the following? ] To provide some control over load balancing, NVA addresses may have an associated priority. Addresses are used in order of priority, with no explicit preference among NVA addresses having the same priority. To provide basic load-balancing among NVAs of equal priorities, NVEs use some randomization input to select among equal-priority NVAs. Such a priority scheme facilitates failover and load balancing, for example, allowing a network operator to specify a set of primary and backup NVAs.

[note: should we support the following? It would presumably add considerable complexity to the NVE.] It may be desirable to have individual NVA addresses responsible for a subset of information about an NV Domain. In such a case, NVEs would use different NVA addresses for obtaining or updating information about particular VNs or TS bindings. A key question with such an approach is how information would be partitioned, and how an NVE could determine which address to use to get the information it needs.

Another possibility is to treat the information on which NVA addresses to use as cached (soft-state) information at the NVEs, so that any NVA address can be used to obtain any information, but NVEs are informed of preferences for which addresses to use for particular information on VNs or TS bindings. That preference information would be cached for future use to improve behavior - e.g., if all requests for a specific subset of VNs are forwarded to a specific NVA component, the NVE can optimize future requests within that subset by sending them directly to that NVA component via its address.

## 8. NVE-to-NVA Protocol

[Note: this and later sections are a bit sketchy and need work. Discussion is encouraged.]

As outlined in Section 4.3, an NVE needs certain information in order to perform its functions. To obtain such information from an NVA, an NVE-to-NVA protocol is needed. The NVE-to-NVA protocol provides two functions. First it allows an NVE to obtain information about the location and status of other TSs with which it needs to communicate. Second, the NVE-to-NVA protocol provides a way for NVEs to provide updates to the NVA about the TSs attached to that NVE (e.g., when a TS attaches or detaches from the NVE), or about communication errors encountered when sending traffic to remote NVEs. For example, an NVE could indicate that a destination it is trying to reach at a destination NVE is unreachable for some reason.

While having a direct NVE-to-NVA protocol might seem straightforward, the existence of existing VM orchestration systems complicates the choices an NVE has for interacting with the NVA.

### 8.1. NVE-NVA Interaction Models

An NVE interacts with an NVA in at least two (quite different) ways:

- o NVEs supporting VMs and hypervisors can obtain necessary information entirely through the hypervisor-facing side of the NVE. Such an approach is a natural extension to existing VM orchestration systems supporting server virtualization because an existing protocol between the hypervisor and VM Orchestration system already exists and can be leveraged to obtain any needed information. Specifically, VM orchestration systems used to create, terminate and migrate VMs already use well-defined (though typically proprietary) protocols to handle the interactions between the hypervisor and VM orchestration system. For such systems, it is a natural extension to leverage the existing orchestration protocol as a sort of proxy protocol for handling

the interactions between an NVE and the NVA. Indeed, existing implementations can already do this.

- o Alternatively, an NVE can obtain needed information by interacting directly with an NVA via a protocol operating over the data center underlay network. Such an approach is needed to support NVEs that are not associated with systems performing server virtualization (e.g., as in the case of a standalone gateway) or where the NVE needs to communicate directly with the NVA for other reasons.

[Note: The following paragraph is included to stimulate discussion, and the WG will need to decide what direction it wants to take.]

The WG The NVO3 architecture should support both of the above models, as in practice, it is likely that both models will coexist in practice and be used simultaneously in a deployment. Existing virtualization environments are already using the first model. But they are not sufficient to cover the case of standalone gateways -- such gateways may not support virtualization and do not interface with existing VM orchestration systems. Also, a hybrid approach might be desirable in some cases where the first model is used to obtain the information, but the latter approach is used to validate and further authenticate the information before using it.

## 8.2. Direct NVE-NVA Protocol

An NVE can interact directly with an NVA via an NVE-to-NVA protocol. Such a protocol can be either independent of the NVA internal protocol, or an extension of it. Using a dedicated protocol provides architectural separation and independence between the NVE and NVA. The NVE and NVA interact in a well-defined way, and changes in the NVA (or NVE) do not need to impact each other. Using a dedicated protocol also ensures that both NVE and NVA implementations can evolve independently and without dependencies on each other. Such independence is important because the upgrade path for NVEs and NVAs is quite different. Upgrading all the NVEs at a site will likely be more difficult in practice than upgrading NVAs because of their large number - one on each end device. In practice, it would be prudent to assume that once an NVE has been implemented and deployed, it may be challenging to get subsequent NVE extensions and changes implemented and deployed, whereas an NVA (and its associated protocols) are more likely to evolve over time as experience is gained from usage and upgrades will involve fewer nodes.

Requirements for a direct NVE-NVA protocol can be found in [I-D.ietf-nvo3-nve-nva-req]

### 8.3. Propagating Information Between NVEs and NVAs

Information flows between NVEs and NVAs in both directions. The NVA maintains information about all VNs in the NV Domain, so that NVEs do not need to do so themselves. NVEs obtain from the NVA information about where a given remote TS destination resides. NVAs in turn obtain information from NVEs about the individual TSs attached to those NVEs.

While the NVA could push information about every virtual network to every NVE, such an approach scales poorly and is unnecessary. In practice, a given NVE will only need and want to know about VNs to which it is attached. Thus, an NVE should be able to subscribe to updates only for the virtual networks it is interested in receiving updates for. The NVO3 architecture supports a model where an NVE is not required to have full mapping tables for all virtual networks in an NV Domain.

Before sending unicast traffic to a remote TS, an NVE must know where the remote TS currently resides. When a TS attaches to a virtual network, the NVE obtains information about that VN from the NVA. The NVA can provide that information to the NVE at the time the TS attaches to the VN, either because the NVE requests the information when the attach operation occurs, or because the VM orchestration system has initiated the attach operation and provides associated mapping information to the NVE at the same time. A similar process can take place with regards to obtaining necessary information needed for delivery of tenant broadcast or multicast traffic.

There are scenarios where an NVE may wish to query the NVA about individual mappings within an VN. For example, when sending traffic to a remote TS on a remote NVE, that TS may become unavailable (e.g., because it has migrated elsewhere or has been shutdown, in which case the remote NVE may return an error indication). In such situations, the NVE may need to query the NVA to obtain updated mapping information for a specific TS, or verify that the information is still correct despite the error condition. Note that such a query could also be used by the NVA as an indication that there may be an inconsistency in the network and that it should take steps to verify that the information it has about the current state and location of a specific TS is still correct.

For very large virtual networks, the amount of state an NVE needs to maintain for a given virtual network could be significant. Moreover, an NVE may only be communicating with a small subset of the TSs on such a virtual network. In such cases, the NVE may find it desirable to maintain state only for those destinations it is actively communicating with. In such scenarios, an NVE may not want to



maintain full mapping information about all destinations on a VN. Should it then need to communicate with a destination for which it does not have mapping information, however, it will need to be able to query the NVA on demand for the missing information on a per-destination basis.

The NVO3 architecture will need to support a range of operations between the NVE and NVA. Requirements for those operations can be found in [I-D.ietf-nvo3-nve-nva-cp-req].

## 9. Federated NVAs

An NVA provides service to the set of NVEs in its NV Domain. Each NVA manages network virtualization information for the virtual networks within its NV Domain. An NV domain is administered by a single entity.

In some cases, it will be necessary to expand the scope of a specific VN or even an entire NV domain beyond a single NVA. For example, multiple data centers managed by the same administrator may wish to operate all of its data centers as a single NV region. Such cases are handled by having different NVAs peer with each other to exchange mapping information about specific VNs. NVAs operate in a federated manner with a set of NVAs operating as a loosely-coupled federation of individual NVAs. If a virtual network spans multiple NVAs (e.g., located at different data centers), and an NVE needs to deliver tenant traffic to an NVE at a remote NVA, it still interacts only with its NVA, even when obtaining mappings for NVEs associated with domains at a remote NVA.

Figure 3 shows a scenario where two separate NV Domains (1 and 2) share information about Virtual Network "1217". VM1 and VM2 both connect to the same Virtual Network 1217, even though the two VMs are in separate NV Domains. There are two cases to consider. In the first case, NV Domain B (NVB) does not allow NVE-A to tunnel traffic directly to NVE-B. There could be a number of reasons for this. For example, NV Domains 1 and 2 may not share a common address space (i.e., require traversal through a NAT device), or for policy reasons, a domain might require that all traffic between separate NV Domains be funneled through a particular device (e.g., a firewall). In such cases, NVA-2 will advertise to NVA-1 that VM1 on Virtual Network 1217 is available, and direct that traffic between the two nodes go through IP-G. IP-G would then decapsulate received traffic from one NV Domain, translate it appropriately for the other domain and re-encapsulate the packet for delivery.



- o Control the quantity and rate of information updates that flow (and must be processed) between different NVAs in different data centers.
- o Control the set of external NVAs (and external sites) a site peers with. A site will only peer with other sites that are cooperating in providing an overlay service.
- o Allow policy to be applied between sites. A site will want to carefully control what information it exports (and to whom) as well as what information it is willing to import (and from whom).
- o Allow different protocols and architectures to be used to for intra- vs. inter-NVA communication. For example, within a single data center, a replicated transaction server using database techniques might be an attractive implementation option for an NVA, and protocols optimized for intra-NVA communication would likely be different from protocols involving inter-NVA communication between different sites.
- o Allow for optimized protocols, rather than using a one-size-fits all approach. Within a data center, networks tend to have lower-latency, higher-speed and higher redundancy when compared with WAN links interconnecting data centers. The design constraints and tradeoffs for a protocol operating within a data center network are different from those operating over WAN links. While a single protocol could be used for both cases, there could be advantages to using different and more specialized protocols for the intra- and inter-NVA case.

#### 9.1. Inter-NVA Peering

To support peering between different NVAs, an inter-NVA protocol is needed. The inter-NVA protocol defines what information is exchanged between NVAs. It is assumed that the protocol will be used to share addressing information between data centers and must scale well over WAN links.

#### 10. Control Protocol Work Areas

The NVO3 architecture consists of two major distinct entities: NVEs and NVAs. In order to provide isolation and independence between these two entities, the NVO3 architecture calls for well defined protocols for interfacing between them. For an individual NVA, the architecture calls for a single conceptual entity that could be implemented in a distributed or replicated fashion. While the IETF may choose to define one or more specific architectural approaches to building individual NVAs, there is little need for it to pick exactly

one approach to the exclusion of others. An NVA for a single domain will likely be deployed as a single vendor product and thus there is little benefit in standardizing the internal structure of an NVA.

Individual NVAs peer with each other in a federated manner. The NVO3 architecture calls for a well-defined interface between NVAs.

Finally, a hypervisor-to-NVE protocol is needed to cover the split-NVE scenario described in Section 4.2.

## 11. NVO3 Data Plane Encapsulation

When tunneling tenant traffic, NVEs add encapsulation header to the original tenant packet. The exact encapsulation to use for NVO3 does not seem to be critical. The main requirement is that the encapsulation support a Context ID of sufficient size [I-D.ietf-nvo3-dataplane-requirements]. A number of encapsulations already exist that provide a VN Context of sufficient size for NVO3. For example, VXLAN [I-D.mahalingam-dutt-dcops-vxlan] has a 24-bit VXLAN Network Identifier (VNI). NVGRE [I-D.sridharan-virtualization-nvgre] has a 24-bit Tenant Network ID (TNI). MPLS-over-GRE provides a 20-bit label field. While there is widespread recognition that a 12-bit VN Context would be too small (only 4096 distinct values), it is generally agreed that 20 bits (1 million distinct values) and 24 bits (16.8 million distinct values) are sufficient for a wide variety of deployment scenarios.

[Note: the following paragraph is included for WG discussion. Future versions of this document may omit this text.]

While one might argue that a new encapsulation should be defined just for NVO3, no compelling requirements for doing so have been identified yet. Moreover, optimized implementations for existing encapsulations are already starting to become available on the market (i.e., in silicon). If the IETF were to define a new encapsulation format, it would take at least 2 (and likely more) years before optimized implementations of the new format would become available in products. In addition, a new encapsulation format would not likely displace existing formats, at least not for years. Thus, there seems little reason to define a new encapsulation. However, it does make sense for NVO3 to support multiple encapsulation formats, so as to allow NVEs to use their preferred encapsulations when possible. This implies that the address dissemination protocols must also include an indication of supported encapsulations along with the address mapping details.

## 12. Operations and Management

The simplicity of operating and debugging overlay networks will be critical for successful deployment. Some architectural choices can facilitate or hinder OAM. Related OAM drafts include [I-D.ashwood-nvo3-operational-requirement].

## 13. Summary

This document provides a start at a general architecture for overlays in NVO3. The architecture calls for three main areas of protocol work:

1. A hypervisor-to-NVE protocol to support Split NVEs as discussed in Section 4.2.
2. An NVE to NVA protocol for address dissemination.
3. An NVA-to-NVA protocol for exchange of information about specific virtual networks between NVAs.

It should be noted that existing protocols or extensions of existing protocols are applicable.

## 14. Acknowledgments

Helpful comments and improvements to this document have come from Lizhong Jin, Dennis (Xiaohong) Qin, Erik Smith, Ziyi Yang and Lucy Yong.

## 15. IANA Considerations

This memo includes no request to IANA.

## 16. Security Considerations

Yep, kind of sparse. But we'll get there eventually. :-)

## 17. Informative References

[I-D.ashwood-nvo3-operational-requirement]  
Ashwood-Smith, P., Iyengar, R., Tsou, T., Sajassi, A., Boucadair, M., Jacquenet, C., and M. Daikoku, "NVO3 Operational Requirements", draft-ashwood-nvo3-operational-requirement-03 (work in progress), July 2013.

[I-D.ietf-nvo3-dataplane-requirements]

Bitar, N., Lasserre, M., Balus, F., Morin, T., Jin, L., and B. Khasnabish, "NVO3 Data Plane Requirements", draft-ietf-nvo3-dataplane-requirements-02 (work in progress), November 2013.

[I-D.ietf-nvo3-framework]

Lasserre, M., Balus, F., Morin, T., Bitar, N., and Y. Rekhter, "Framework for DC Network Virtualization", draft-ietf-nvo3-framework-05 (work in progress), January 2014.

[I-D.ietf-nvo3-nve-nva-cp-req]

Kreeger, L., Dutt, D., Narten, T., and D. Black, "Network Virtualization NVE to NVA Control Protocol Requirements", draft-ietf-nvo3-nve-nva-cp-req-01 (work in progress), October 2013.

[I-D.ietf-nvo3-overlay-problem-statement]

Narten, T., Gray, E., Black, D., Fang, L., Kreeger, L., and M. Napierala, "Problem Statement: Overlays for Network Virtualization", draft-ietf-nvo3-overlay-problem-statement-04 (work in progress), July 2013.

[I-D.kreeger-nvo3-hypervisor-nve-cp]

Kreeger, L., Narten, T., and D. Black, "Network Virtualization Hypervisor-to-NVE Overlay Control Protocol Requirements", draft-kreeger-nvo3-hypervisor-nve-cp-01 (work in progress), February 2013.

[I-D.mahalingam-dutt-dcops-vxlan]

Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", draft-mahalingam-dutt-dcops-vxlan-08 (work in progress), February 2014.

[I-D.sridharan-virtualization-nvgre]

Sridharan, M., Greenberg, A., Wang, Y., Garg, P., Venkataramiah, N., Duda, K., Ganga, I., Lin, G., Pearson, M., Thaler, P., and C. Tumuluri, "NVGRE: Network Virtualization using Generic Routing Encapsulation", draft-sridharan-virtualization-nvgre-04 (work in progress), February 2014.

- [IEEE-802.1Q] IEEE 802.1Q-2011, , "IEEE standard for local and metropolitan area networks: Media access control (MAC) bridges and virtual bridged local area networks," , August 2011.
- [RFC0826] Plummer, D., "Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware", STD 37, RFC 826, November 1982.
- [RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, February 2006.
- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

## Appendix A. Change Log

### A.1. Changes From draft-ietf-nvo3-arch-00 to -01

1. Miscellaneous text/section additions, including:
  - \* New section on VLAN tag Handling (Section 3.1.1).
  - \* New section on tenant VLAN handling in Split-NVE case (Section 4.2.1).
  - \* New section on TTL handling (Section 3.1.2).
  - \* New section on multi-homing of NVEs (Section 4.4).
  - \* 2 paragraphs new text describing L2/L3 Combined service (Section 3.1).
  - \* New section on VAPs (and error handling) (Section 4.5).
  - \* New section on ARP and ND handling (Section 5.5)
  - \* New section on NVE-to-NVE interactions (Section 6)
2. Editorial cleanups from careful review by Erik Smith, Ziye Yang.
3. Expanded text on Distributed Inter-VN Gateways.

## A.2. Changes From draft-narten-nvo3 to draft-ietf-nvo3

1. No changes between draft-narten-nvo3-arch-01 and draft-ietf-nvoe-arch-00.

## A.3. Changes From -00 to -01 (of draft-narten-nvo3-arch)

1. Editorial and clarity improvements.
2. Replaced "push vs. pull" section with section more focused on triggers where an event implies or triggers some action.
3. Clarified text on co-located NVE to show how offloading NVE functionality onto adapters is desirable.
4. Added new section on distributed gateways.
5. Expanded Section on NVA external interface, adding requirement for NVE to support multiple IP NVA addresses.

## Authors' Addresses

David Black  
EMC

Email: david.black@emc.com

Jon Hudson  
Brocade  
120 Holger Way  
San Jose, CA 95134  
USA

Email: jon.hudson@gmail.com

Lawrence Kreeger  
Cisco

Email: kreeger@cisco.com

Marc Lasserre  
Alcatel-Lucent

Email: marc.lasserre@alcatel-lucent.com



Thomas Narten  
IBM

Email: [narten@us.ibm.com](mailto:narten@us.ibm.com)

Internet Engineering Task Force  
Internet-Draft  
Intended status: Informational  
Expires: March 29, 2014

E. Gray, Ed.  
Ericsson  
N. Bitar  
Verizon  
X. Chen  
Huawei Technologies  
M. Lasserre  
Alcatel-Lucent  
T. Tsou  
Huawei Technologies (USA)  
September 25, 2013

NVO3 Gap Analysis - Requirements Versus Available Technology Choices  
draft-ietf-nvo3-gap-analysis-00

Abstract

This document evaluates candidate protocols against the NVO3 requirements. Gaps are identified and further work recommended.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 29, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Terminology and Conventions . . . . .	3
2.1. Requirements Language . . . . .	3
2.2. Conventions . . . . .	3
2.3. Terms and Abbreviations . . . . .	3
3. Operational Requirements . . . . .	4
4. Management Requirements . . . . .	4
5. Control Plane Requirements . . . . .	4
5.1. Overall Control-Plane Requirements . . . . .	5
5.2. VM-to-NVE Specific Control-Plane Requirements . . . . .	7
6. Data Plane Requirements . . . . .	9
7. Summary and Conclusions . . . . .	14
8. Acknowledgements . . . . .	14
9. IANA Considerations . . . . .	15
10. Security Considerations . . . . .	15
11. References . . . . .	15
11.1. Normative References . . . . .	15
11.2. Informative References . . . . .	17
Authors' Addresses . . . . .	17

## 1. Introduction

The initial charter of the NV03 Working Group requires it to identify any gaps between the requirements identified and available technology solutions as a prerequisite to rechartering or concluding the Working Group (if no gaps exist). This document is intended to provide the required gap analysis.

This document provides a tabulation of candidate solutions and their ability to satisfy each requirement identified by the Working Group.

Areas of work are identified where further work is required to ensure that the requirements are met.

The major areas covered in this document include:

- o Operational Requirements  
[I-D.ashwood-nvo3-operational-requirement]
- o Management Requirements (TBD)

- o Control (Plane) Requirements [I-D.kreeger-nvo3-overlay-cp]
- o Dataplane Requirements [I-D.ietf-nvo3-dataplane-requirements]

Since the Working Group has yet to complete (and in some cases adopt) documents describing requirements for some of these areas, not all areas are complete in the present version of this document.

The initial candidate technologies are:

- o NVGRE [I-D.sridharan-virtualization-nvgre],
- o VxLAN [I-D.mahalingam-dutt-dcops-vxlan],
- o L2VPN: VPLS [RFC4761][RFC4762] and EVPN [I-D.ietf-l2vpn-evpn], and
- o L3VPN [RFC4365].

## 2. Terminology and Conventions

### 2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

### 2.2. Conventions

In sections providing analysis of requirements defined in referenced documents, section numbers from each referenced document are used as they were listed in that document.

In order to avoid confusing those section numbers with the section numbering in this document, the included numbering is parenthesized.

L2VPN is represented (in tables and analysis, as a technology) by the two differing approaches: VPLS and EVPN.

### 2.3. Terms and Abbreviations

This document uses terms and acronyms defined in [RFC3168], [I-D.ietf-nvo3-framework], [I-D.ietf-nvo3-dataplane-requirements], [I-D.kreeger-nvo3-hypervisor-nve-cp] and [I-D.kreeger-nvo3-overlay-cp]. Acronyms are included here for convenience but are meant to remain aligned with definitions in the references included.

ECN: Explicit Congestion Notification [RFC3168]

NVA: Network Virtualization Authority [I-D.kreeger-nvo3-overlay-cp]

NVE: Network Virtualization Edge [I-D.ietf-nvo3-framework]

VAP: Virtual Access Point [I-D.ietf-nvo3-dataplane-requirements]

VNI: Virtual Network Instance [I-D.ietf-nvo3-framework]

VNIC: Virtual Network Interface Card (NIC)  
[I-D.kreeger-nvo3-hypervisor-nve-cp]

VNID: Virtual Network Identifier [I-D.kreeger-nvo3-overlay-cp]

This document uses the following additional general terms and abbreviations:

DSCP: Differentiated Services Code-Point

ECMP: Equal Cost Multi-Path

L2VPN: Layer 2 Virtual Private Network

L3VPN: Layer 3 Virtual Private Network

NVO3: Network Virtualization Overlay over L3

VM: Virtual Machine

VN: Virtual Network

### 3. Operational Requirements

TBD

### 4. Management Requirements

TBD

### 5. Control Plane Requirements

The NVO3 Problem Statement [I-D.ietf-nvo3-overlay-problem-statement], describes 3 categories of control functions:

1. Control functions associated with implementing the Network Virtualization Authority (e.g. - signaling and control required for interactions between multiple NVA devices).

2. Control functions associated with interactions between an NVA and a Network Virtualization Edge (NVE).
3. Control functions associated with attaching and detaching a Virtual Machine (VM) from a particular Virtual Network Instance (VNI).

As sometimes happens, there is not a 1:1 mapping of the work areas defined in [I-D.ietf-nvo3-overlay-problem-statement] and requirements documents intended to address the problems that have been identified there.

Current control-plane requirement documents include the following:

- o Overall control-plane requirements [I-D.kreeger-nvo3-overlay-cp]
- o Control-plane requirements specific to VM-to-NVE interactions [I-D.kreeger-nvo3-hypervisor-nve-cp]

#### 5.1. Overall Control-Plane Requirements

In this section, numbering of requirement headings corresponds to section numbering in [I-D.kreeger-nvo3-overlay-cp].

##### (3.1) Inner to Outer Address Mapping

The requirements document [I-D.kreeger-nvo3-overlay-cp] states that avoiding the need to "flood" traffic to support learning of mapping information from the data-plane is a goal of NVO3 candidate technological approaches.

For each candidate technology, (how) is the mapping of header information present in tenant traffic mapped to corresponding header information to be used in overlay encapsulation (this includes addresses, context identification, etc.) determined?

Supported Approach	NVGRE	VxLAN	VPLS	EVPN	L3VPN
Control Protocol Acquisition?					
- - -	- - -	- - -	- - -	- - -	- - -
Data-Plane Learning?					

Table 1: Inner:Outer Address Mapping

##### (3.2) Underlying Network Multi-Destination Address(es)

The requirements document [I-D.kreeger-nvo3-overlay-cp] lists 3 approaches that may be used to deliver traffic to multiple destinations in an overlay virtual network:

1. Use the capabilities of the underlay network.
2. Require a sending NVE to replicate traffic.
3. Use a replication service provided within the overlay network.

For each delivery approach, it may be necessary to map specific multipoint (e.g. - broadcast, unknown destination or multicast) traffic to (for instance) addresses used to deliver this traffic via the underlay network.

For each technological approach, which delivery approaches are supported and does the technology provide a method by which an NVE needing to send multi-destination traffic can determine to what address, or addresses to which to send this traffic?

Supported Approach	NVGRE	VxLAN	VPLS	EVPN	L3VPN
Underlay Network Capability					
NVE Sender Replication					
Replication Service					

Table 2: Multi-Destination Delivery

### (3.3) VN Connect/Disconnect Notification

The requirements document [I-D.kreeger-nvo3-overlay-cp] states as an assumption that a mechanism exists in the overlay technology by which an NVE is notified of Tenant Systems attaching and detaching from a specific Virtual Network (VN).

For each candidate technology, does the technology currently support these functions?

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
Connect Notification					

Disconnect Notification						
-------------------------	--	--	--	--	--	--

Table 3: Connect/Disconnect Notification

## (3.4) VN Name to VNID Mapping

The requirements document [I-D.kreeger-nvo3-overlay-cp] concludes that having a means to map for a "VN Name to a "VN ID" may be useful.

For each technological approach we are considering, is this function currently available?

Function	NVGRE	VxLAN	VPLS	EVPN	L3VPN
VN-Name:VN-ID Mapping					

Table 4: VN Name to VN ID Mapping

## 5.2. VM-to-NVE Specific Control-Plane Requirements

In this section, numbering of requirement headings corresponds to section numbering in [I-D.kreeger-nvo3-hypervisor-nve-cp].

## (4.1) VN Connect/Disconnect

The requirements document [I-D.kreeger-nvo3-hypervisor-nve-cp] states as a requirement that a mechanism must exist by which an NVE is notified when an end device requires a connection, or no longer requires a connection, to a specific Virtual Network (VN).

The requirements document further states as a requirement that the mechanism(s) used in a candidate technological approach must provide a local indicator (e.g. - 802.1Q tag) that the end device will use in sending traffic to, or receiving traffic from, the NVE (where that traffic is associated with the connected VN).

As an additional related requirement, the requirements document states that the NVE - once notified of a connection to a VN (by VN Name), needs to have a means for getting associated VN context information from the NVA.

For each candidate technology, does the technology currently support these functions?



Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
Connect Notification					
Local VN Indicator					
VN Name to VN Context Mapping					
Disconnect Notification					

Table 5: VN Connect/Disconnect

## (4.2) VNIC Address Association

The requirements document [I-D.kreeger-nvo3-hypervisor-nve-cp] lists two approaches for acquiring VNIC address association information:

1. Data Plane Learning (i.e. - by inspecting source addresses in traffic received from an end device).
2. Explicit signaling from the end device when a specific VNIC address is to be associated with a tenant system.

Supported Approaches	NVGRE	VxLAN	VPLS	EVPN	L3VPN
Data Plane Learning					
Explicit Signaling					

Table 6: VNIC Address Association

## (4.3) VNIC Address Disassociation

TBD

## (4.4) VNIC Shutdown/Startup/Migration

TBD

## (4.5) VN Profile

TBD

## 6. Data Plane Requirements

In this section, numbering of requirement headings corresponds to section numbering in [I-D.ietf-nvo3-dataplane-requirements].

### (3.1) Virtual Access Points (VAPs)

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
MUST support VAP identification					
1) Local interface	YES				
2) Local interface + fields in frame header	YES				

Table 7: VAP Identification Requirements

### (3.2) Virtual Network Instance (VNI)

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
VAP are associated with a specific VNI at service instantiation time.	YES				

Table 8: VAP-VNI Association

#### (3.2.1) L2 VNI

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
L2 VNI MUST provide an emulated Ethernet multipoint service as if Tenant Systems are interconnected by a bridge (but instead by using a set of NV03 tunnels).					

Loop avoidance capability MUST be provided. - - -	- - -	- - -	- - -	- - -	- - -
In the absence of a management or control plane, data plane learning MUST be used to populate forwarding tables. - - -	- - -	- - -	- - -	- - -	- - -
When flooding is required, either to deliver unknown unicast, or broadcast or multicast traffic, the NVE MUST either support ingress replication or multicast. - - -	- - -	- - -	- - -	- - -	- - -
In this latter case, the NVE MUST be able to build at least a default flooding tree per VNI.					

Table 9: L2 VNI Service

## (3.2.2) L3 VNI

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
L3 VNIs MUST provide virtualized IP routing and forwarding. - - -	- - -	- - -	- - -	- - -	- - -
L3 VNIs MUST support per- tenant forwarding instance with IP addressing isolation and L3 tunneling for interconnecting instances of the same VNI on NVEs.					

Table 10: L3 VNI Service

## (3.3.1) NVO3 overlay header

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
An NVO3 overlay header MUST be included after the underlay tunnel header when forwarding tenant traffic.	YES	YES	YES	YES	YES

Table 11: Overlay Header

## (3.3.1.1) Virtual Network Context Identification

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
The overlay encapsulation header MUST contain a field which allows the encapsulated frame to be delivered to the appropriate virtual network endpoint by the egress NVE.	YES	YES	YES	YES	YES

Table 12: Virtual Network Context Identification

## (3.3.1.2) Service QoS identifier

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
Traffic flows originating from different applications could rely on differentiated forwarding treatment to meet end-to-end availability and performance objectives.	NO				

Table 13: QoS Service Identification

## (3.3.2.1) LAG and ECMP

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
For performance reasons, multipath over LAG and ECMP paths SHOULD be supported.	YES				

Table 14: Multipath Support

## (3.3.2.2) DiffServ and ECN marking

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
[RFC2983] defines two modes for mapping the DSCP markings from inner to outer headers and vice versa. Both models SHOULD be supported.	NO				
---	---	---	---	---	---
ECN marking MUST be performed according to [RFC6040] which describes the correct ECN behavior for IP tunnels.	NO			-	

Table 15: DSCP and ECN Marking

## (3.3.2.3) Handling of broadcast, unknown unicast, and multicast traffic

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
NV03 data plane support for either ingress replication or point-to-multipoint tunnels is required to send traffic destined to multiple locations on a per-VNI basis (e.g. L2/L3 multicast traffic, L2 broadcast and unknown	YES	YES	YES	YES	YES

unicast traffic).					
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+

Table 16: Handling of Broadcast, Unknown Unicast, and Multicast Traffic

(3.4) External NVO3 connectivity

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
NVO3 services MUST interoperate with current VPN and Internet services. This may happen inside one DC during a migration phase or as NVO3 services are delivered to the outside world via Internet or VPN gateways.	YES				

Table 17: Interoperation

(3.5) Path MTU

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
Classical ICMP-based MTU Path Discovery ([RFC1191], [RFC1981]) or Extended MTU Path Discovery techniques such as defined in [RFC4821].	NO				
Segmentation and reassembly support from the overlay layer operations without relying on the Tenant Systems to know about the end-to-end MTU.	YES				

Table 18: Path MTU

## (3.7) NVE Multi-Homing Requirements

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
Multi-homing techniques SHOULD be used to increase the reliability of an NV03 network.	NO				

Table 19: Multihoming

## (3.8) OAM

Requirement	NVGRE	VxLAN	VPLS	EVPN	L3VPN
NVE MAY be able to originate/terminate OAM messages for connectivity verification, performance monitoring, statistic gathering and fault isolation. Depending on configuration, NVEs SHOULD be able to process or transparently tunnel OAM messages, as well as supporting alarm propagation capabilities.	NO				

Table 20: OAM Messaging

## 7. Summary and Conclusions

TBD

## 8. Acknowledgements

The Authors would like to acknowledge the technical contributions of Florin Balus, Luyuan Fang, Sue Hares, Wim Henderickx, Yuichi Ikejiri, Rangaraju Iyengar, Mircea Pisica, Evelyn Roch, Ali Sajassi, Peter Ashwood-Smith and Lucy Yong as well as the initial help in editing the XML source for the document from Tom Taylor.

## 9. IANA Considerations

This memo includes no request to IANA.

## 10. Security Considerations

Security considerations of the requirements documents referenced by this analysis document apply.

## 11. References

### 11.1. Normative References

- [I-D.ashwood-nvo3-operational-requirement]  
Ashwood-Smith, P., Iyengar, R., Tsou, T., Sajassi, A., Boucadair, M., Jacquenet, C., and M. Daikoku, "NVO3 Operational Requirements", draft-ashwood-nvo3-operational-requirement-03 (work in progress), July 2013.
- [I-D.ietf-l2vpn-evpn]  
Sajassi, A., Aggarwal, R., Henderickx, W., Balus, F., Isaac, A., and J. Uttaro, "BGP MPLS Based Ethernet VPN", draft-ietf-l2vpn-evpn-04 (work in progress), July 2013.
- [I-D.ietf-nvo3-dataplane-requirements]  
Bitar, N., Lasserre, M., Balus, F., Morin, T., Jin, L., and B. Khasnabish, "NVO3 Data Plane Requirements", draft-ietf-nvo3-dataplane-requirements-01 (work in progress), July 2013.
- [I-D.ietf-nvo3-framework]  
Lasserre, M., Balus, F., Morin, T., Bitar, N., and Y. Rekhter, "Framework for DC Network Virtualization", draft-ietf-nvo3-framework-03 (work in progress), July 2013.
- [I-D.ietf-nvo3-overlay-problem-statement]  
Narten, T., Gray, E., Black, D., Fang, L., Kreeger, L., and M. Napierala, "Problem Statement: Overlays for Network Virtualization", draft-ietf-nvo3-overlay-problem-statement-04 (work in progress), July 2013.
- [I-D.kreeger-nvo3-hypervisor-nve-cp]  
Kreeger, L., Narten, T., and D. Black, "Network Virtualization Hypervisor-to-NVE Overlay Control Protocol Requirements", draft-kreeger-nvo3-hypervisor-nve-cp-01 (work in progress), February 2013.
- [I-D.kreeger-nvo3-overlay-cp]



Kreeger, L., Dutt, D., Narten, T., Black, D., and M. Sridharan, "Network Virtualization Overlay Control Protocol Requirements", draft-kreeger-nvo3-overlay-cp-04 (work in progress), June 2013.

[I-D.mahalingam-dutt-dcops-vxlan]

Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", draft-mahalingam-dutt-dcops-vxlan-04 (work in progress), May 2013.

[I-D.sridharan-virtualization-nvgre]

Sridharan, M., Greenberg, A., Wang, Y., Garg, P., Venkataramiah, N., Duda, K., Ganga, I., Lin, G., Pearson, M., Thaler, P., and C. Tumuluri, "NVGRE: Network Virtualization using Generic Routing Encapsulation", draft-sridharan-virtualization-nvgre-03 (work in progress), August 2013.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, November 1990.

[RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery for IP version 6", RFC 1981, August 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2983] Black, D., "Differentiated Services and Tunnels", RFC 2983, October 2000.

[RFC4365] Rosen, E., "Applicability Statement for BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4365, February 2006.

[RFC4761] Kompella, K. and Y. Rekhter, "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling", RFC 4761, January 2007.

[RFC4762] Lasserre, M. and V. Kompella, "Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling", RFC 4762, January 2007.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, March 2007.

[RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", RFC 6040, November 2010.

## 11.2. Informative References

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.

## Authors' Addresses

Eric Gray (editor)  
Ericsson  
120 Morris Avenue  
Pitman, New Jersey 08071  
USA

Email: [eric.gray@ericsson.com](mailto:eric.gray@ericsson.com)

Nabil Bitar  
Verizon  
40 Sylvan Road  
Waltham, Massachusetts 02145  
USA

Email: [nabil.bitar@verizon.com](mailto:nabil.bitar@verizon.com)

Xiaoming Chen  
Huawei Technologies

Email: [ming.chen@huawei.com](mailto:ming.chen@huawei.com)

Marc Lasserre  
Alcatel-Lucent

Email: [marc.lasserre@alcatel-lucent.com](mailto:marc.lasserre@alcatel-lucent.com)

Tina Tsou  
Huawei Technologies (USA)  
2330 Central Expressway  
Santa Clara, California 95050  
USA

Phone: +1 408 330 4424  
Email: [Tina.Tsou.Zouting@huawei.com](mailto:Tina.Tsou.Zouting@huawei.com)  
URI: <http://tinatsou.weebly.com/contact.html>

Network Working Group  
Internet-Draft  
Intended status: Experimental  
Expires: July 28, 2014

S. Hartman  
Painless Security  
D. Zhang  
Huawei  
M. Wasserman  
Painless Security  
January 24, 2014

Security Requirements of NVO3  
draft-ietf-nvo3-security-requirements-02

Abstract

The draft describes a list of essential requirements in order to benefit the design of NOV3 security solutions. In addition, this draft introduces the candidate techniques which could be used to construct a security solution fulfilling these security requirements.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 28, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	3
3. NVO3 Overlay Architecture . . . . .	4
4. Threat Model . . . . .	4
4.1. Capabilities of Outsiders . . . . .	5
4.2. Capabilities of Insiders . . . . .	5
4.3. Capabilities of Malicious TSes . . . . .	5
4.4. Security Issues In Scope and Out of Scope . . . . .	6
5. Security Requirements . . . . .	7
5.1. Control/Data Plane of NVO3 Overlay . . . . .	7
5.1.1. NVE-NVA Control Plane . . . . .	7
5.1.2. NVA-NVA Control Plane . . . . .	9
5.1.3. NVE-NVE Data Plane . . . . .	10
5.2. Control/Data Plane between NVEs and Hypervisors . . . . .	11
5.2.1. Distributed Deployment of NVE and Hypervisor . . . . .	12
6. Candidate Techniques . . . . .	14
6.1. Entity Authentication . . . . .	14
6.2. Packet Level Security . . . . .	15
6.3. Authorization . . . . .	15
7. IANA Considerations . . . . .	15
8. Security Considerations . . . . .	15
8.1. Automated Key Management in NVO3 . . . . .	15
8.2. Issues not Discussed . . . . .	16
9. Acknowledgements . . . . .	16
10. References . . . . .	16
10.1. Normative References . . . . .	16
10.2. Informative References . . . . .	17
Authors' Addresses . . . . .	18

## 1. Introduction

Security is a key issue which needs to be considered during the design of a data center network. This document discusses the security risks that a NVO3 network may encounter and tries to provide a list of essential security requirements that a NVO3 network needs to fulfill. In addition, this draft introduces the candidate

techniques which could be potentially used to construct a security solution fulfilling the security requirements.

The remainder of this document is organized as follows. Section 2 introduces several key terms used in this memo. Section 3 gives a brief introduction of the NVO3 network architecture. Section 4 discusses the attack model of this work. Section 5 provides a list of security requirements as well as the associated justifications. In Section 6, the candidate techniques are introduced.

## 2. Terminology

This document uses the same terminology as found in the NVO3 Framework document [I-D.ietf-nvo3-framework] and [I-D.kreeger-nvo3-hypervisor-nve-cp]. Some of the terms defined in the framework document have been repeated in this section for the convenience of the reader, along with additional terminology that is used by this document.

**Tenant System (TS):** A physical or virtual system that can play the role of a host, or a forwarding element such as a router, switch, firewall, etc. It belongs to a single tenant and connects to one or more VNs of that tenant.

**End System (ES):** An end system of a tenant, which can be, e.g., a virtual machine (VM), a non-virtualized server, or a physical appliance. A TS is attached to a Network Virtualization Edge (NVE) node.

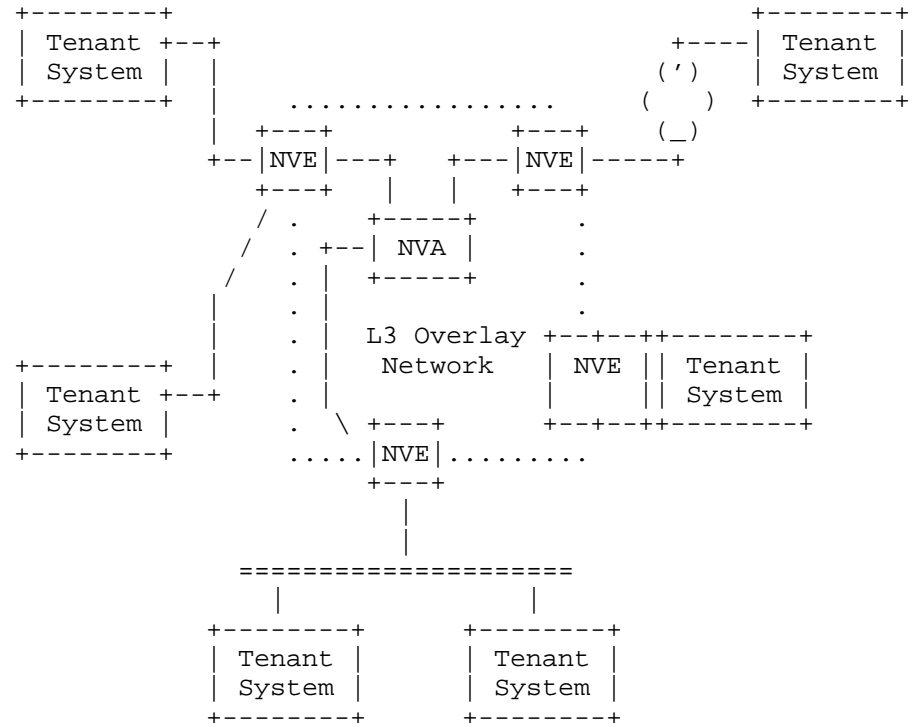
**Network Virtualization Edge (NVE):** An NVE implements network virtualization functions that allow for L2/L3 tenant separation and tenant-related control plane activity. An NVE contains one or more tenant service instances whereby a TS interfaces with its associated instance. The NVE also provides tunneling overlay functions.

**Virtual Network (VN):** This is a virtual L2 or L3 domain that belongs to a tenant.

**Network Virtualization Authority (NVA).** A back-end system that is responsible for distributing and maintaining the mapping information for the entire overlay system.

**NVO3 device:** In this memo, the devices (e.g., NVE and NVA) work cooperatively to provide NVO3 overlay functionalities are called as NVO3 devices.

### 3. NVO3 Overlay Architecture



This figure illustrates a simple nov3 overlay example where NVEs provide a logical L2/L3 interconnect for the TSeS that belong to a specific tenant network over L3 networks. A packet from a tenant system is encapsulated when they reach the ingress NVE. Then encapsulated packet is then sent to the remote NVE through a proper tunnel. When reaching the egress NVE of the tunnel, the packet is decapsulated and forwarded to the target tenant system. The address advertisements and tunnel mappings are distributed to the NVEs by a logically centralized server (i.e., NVA).

### 4. Threat Model

To benefit describing the threats a NVO3 network may have to face, in this work, attacks are classified into three categories: the attacks from compromised NVO3 devices (inside attacks), the attacks from compromised tenant systems, and the attacks from underlying networks (outside attacks).

The adversaries performing the first type of attack are called as insiders or inside attackers because they need to get certain

privileges in changing the configuration or software of NVO3 devices beforehand and initiate the attacks within the overlay security perimeter. In the second type of attack, an attacker has got certain privileges in changing the configuration or software of tenant systems (e.g., hypervisors or virtual machines) and attempts to manipulate the controlled tenant systems to interfere with the normal operations of the NVO3 overlay. The third type of attack is referred to as the outside attack since adversaries do not have to obtain any privilege on the NVO3 devices or tenant systems in advance in order to perform this type attack, and thus the adversaries performing outside attacks are called as outside attackers or outsiders.

#### 4.1. Capabilities of Outsiders

In practice, an outside attacker may perform attacks by intercepting packets, deleting packets, and/or inserting bogus packets. With a successful outside attack, an attacker may be able to:

1. Analyze the traffic pattern within the network,
2. Disrupt the network connectivity or degrade the network service quality, or
3. Access the contents of the data/control packets which are not properly encrypted.

#### 4.2. Capabilities of Insiders

Besides intercepting packets, deleting packets, and/or inserting bogus packets, an inside attacker may use already obtained privilege to,

1. Interfere with the normal operations of the overlay as a legal NVO3 device, by sending packets containing invalid information or with improper frequencies,
2. Perform spoofing attacks and impersonate another legal NVO3 device to communicate with victims using the cryptographic information it obtained, and
3. Access the contents of the data/control packets if they are encrypted with the keys held by the attacker.

#### 4.3. Capabilities of Malicious TSes

It is assumed that the attacker performing attacks from compromised TSes is able to intercept packets, delete packets, and/or insert

bogus packets. In addition, after compromising a TS, an attacker may be able to:

1. Interfere with the normal operations of the overlay as a legal TS, by sending packets containing invalid information or with improper frequencies to NVEs,
2. Perform spoofing attacks and impersonate another legal TS or NVE to communicate with victims (other legal NVEs or Tses) using the cryptographic information it obtained, and
3. Access the contents of the data/control packets if they are encrypted with the keys held by the attacker.

#### 4.4. Security Issues In Scope and Out of Scope

During the specification of security requirements, the following security issues needs to be considered:

1. A underlying network connecting NOV3 devices (NVEs and NVAs) is relatively secure if it is located within a data center and cannot be directly accessed by any tenants or outsiders. However, a NVO3 overlay for virtual data center may scatter across different geographically distributed sites which are connected through the public Internet. In this case, outside attacks may be raised from the underlying network connecting NVO3 devices.
2. During the design of a security solution for a NVO3 network, the attacks raised from compromised NVEs and hypervisors needs to be considered.
3. It is reasonable to consider the conditions where the network connecting Tses and NVEs is accessible to outside attackers.

The following issues are out of scope of consideration in this document:

1. In this memo it is assumed that security protocols, algorithms, and implementations provide the security properties for which they are designed; attacks depending on a failure of this assumption are out of scope. For instance, an attack caused by a weakness in a cryptographic algorithm is out of scope, while an attack caused by failure to use confidentiality when confidentiality is a security requirement is in scope.
2. An attacker controlling an underlying network device may break the communication of the overlays by discarding or delaying the



delivery of the packets passing through it. This type of attack is out of scope of this memo.

3. NVAs are centralized servers and play a critical role in NVO3 overlays. A NVE will believe in the mapping information obtained from its NVA. After compromising a NVA, the attacker can distribute bogus mapping information to NVEs under the management of NVA. This work does not consider how to deal with this problem.

## 5. Security Requirements

### 5.1. Control/Data Plane of NVO3 Overlay

In this section, the security requirements associated with the NVE-NVA control plane, the NVA-NVA control plane, and the NVE-NVE data plane are proposed.

#### 5.1.1. NVE-NVA Control Plane

In a NVE-NVA control plane, it is assumed that a NVE only exchanges control traffics with its NVA using unicast.

REQ 1: The security solution for NVO3 SHOULD enable two NVO3 devices to mutually authenticate each other before they exchange any control packets.

Entity authentication can protect a network device against imposter attacks and then reduce the risk of DoS attacks and man-in-the-middle attacks. In addition, a successful authentication normally results in the distribution key materials for the security protection for subsequent communications.

REQ 2: The security solution of NVO3 MUST be able to provide integrity protection, replay protection, and packet origin authentication for the control packets.

Unlike entity authentication, message authentication is performed on each incoming packet. Through message authentication, the NVO3 device receiving a control packet can verify whether the packet is generated by a legitimate NVO3 device, is not antique, and is not tampered during transportation. In addition, with the support of properly distributed keys, the packet level protection can also benefit the detection of spoofing attacks raised from insiders.

REQ 3: The security solution of a NVO3 network MAY provide confidentiality protection for the control packets.

On many occasions, the control packets can be transported in plaintext. However, under the circumstances where some information contained within the control packets is considered to be sensitive or valuable, the information needs to be encrypted especially when the underlying network is not secure. Note that encryption will impose additional overhead in processing control packets and make NVAs more vulnerable to DoS/DDoS attacks.

REQ 4: Before accepting a control packet, a NOV3 device receiving the packet MUST be able to verify whether the packet comes from one who has the privilege to send that packet.

When receiving a control packet, besides authentication, authorization needs to be carried out by the receiver to identify the role that the packet sender acts as in the overlay and then assess the sender's privileges. If a compromised NVE tries to illegally elevate its privilege (e.g., using its credentials to communicate with other NVEs as a NVA, or attempting to access the mapping information of the VNs which it is not authorized to serve), it will be detected and rejected.

REQ 5: The security solution of NVO3 SHOULD be able to provide distinct keys to protect the control traffics exchanged between a NVA and different NVEs respectively.

During the exchange of control packets, keys are critical in authenticating the packet senders. The purpose of this requirement is to provide a basic capability to confine the damage caused by inside attacks. After compromising a NVE, an attacker will not be able to use the keys it obtained to breach the security of the control traffics exchanged between the NVA and other NVEs.

In a NVO3 overlay, NVAs can be the valuable targets of DoS/DDoS attacks, and large amount of NVEs can be potentially used as reflectors in reflection attacks. Therefore, the DoS/DDoS risks needs be considered during designing the control planes for NOV3. The following two requirements are used to benefit the migration of DoS/DDoS issue.

REQ 6: A NVO3 device MUST send its control packets with limited frequencies.

Without this limitation, an attacker can attempt to perform DDoS attacks to exhaust the limited computing and memory resources of a NVA by manipulating the NVEs attached to the NVA to generate a significant member of mapping queries in a short period.

REQ 7: The amplification effect SHOULD be avoided

If in certain conditions the responses generated by a NVE are much longer than the received requests, the NVE may be taken advantage of by an attacker as a reflector to carry out DDoS attacks. Specifically, the attacker can concurrently send out a large amount of spoofed short requests to multiple NVEs with the source address of a victim (e.g., a NVA). The responses generated by the NVEs will be forwarded to the victim and overwhelm the victim's processing capability.

#### 5.1.2. NVA-NVA Control Plane

Multiple NVAs may be deployed in a NVO3 overlay for better scalability and fault tolerance capability. The NVAs may use unicast and/or multicast to exchange signaling packets within the control plane.

Except the key deployment requirement (REQ 5), all the other requirements in the NVE-NVA control plane (REQs 1,2,3,4, 6, and 7) are applicable in the NVA-NVA control plane as well. Before two NVA communicate with each other, they should be able to mutually authenticated. In addition, message authentication can help a NVO3 device to verify the authenticity of the received packets, and the sensitive information in the control packets need to be encrypted. Authorization is important to filter the invalid control packets and any un-privileged requests. Moreover, the approach to mitigating DoS/DDoS attacks needs to be considered in the control plane protocols.

The key deployment requirements for the NVA-NVA control plane are described as follows:

REQ 8: The security solution of NVO3 SHOULD be able to provide different keys to protect the unicast control traffics exchanged between different NVAs respectively.

The purpose of this requirement is to provide a basic capability to confine the damage caused by compromised key. The compromise of a key will not affect the traffics protected by other keys.

REQ 9: If there are multicast packets, the security solution of NVO3 SHOULD be able to assign distinct cryptographic group keys to protect the multicast packets exchanged among the NVAs within different multicast groups.

In order to provide an essential packet level security protection specified in REQs 2 and 3, at least a group key may need to be

shared among the NVEs in a same multicast group. It is recommended to use different keys for different multicast groups.

#### 5.1.3. NVE-NVE Data Plane

As specified in [I-D.ietf-nvo3-framework], a NVO3 overlay needs to generate tunnels between NVEs for data packet transportation. When a data packet reaches the boundary of a overlay, the ingress NVE will encapsulate the packet and forward it to the destination egress NVE through a proper tunnel.

REQ 10: The security solution for NVO3 SHOULD enable two NVEs to mutually authenticate each other before establishing a tunnel connecting them for data transportation.

This entity authentication requirement is used to protect a NVE against imposter attacks. Also, this requirement can help guarantee a data tunnel is generated between two proper NVEs and reduce the risk of man-in-the-middle attacks.

In order to protect the data packets transported over the overlay against the attacks raised from the underlying network, the NVO3 overlay needs to provide essential security protection for data packets.

REQ 11: The security solution of NVO3 MUST be able to provide integrity protection, replay protection, and packet origin authentication for data traffics exchanged between NVEs.

This requirement is used to prevent an attacker who has compromised a underlying network devices on the path from replaying antique packets or injecting bogus data packets without being detected.

REQ 12: The security solution of NVO3 MAY provide confidentiality protection for data traffics exchanged between NVEs.

If the data traffics from the TSes are sensitive, they need to be encrypted when being transported within the overlay. Otherwise, encryption will be unnecessary. In addition, in practice, tenants may also select to encrypt their sensitive data during transportation. Therefore this confidentiality requirement for data plane is then not as crucial as the integrity requirement.

REQ 13: The security solution of NVO3 SHOULD be able to assign different cryptographic keys to protect the unicast tunnels between NVEs respectively.

This requirement is used to confine the damage caused by inside attacks. When different tunnels secured with different keys, the compromise of a key in a tunnel will not affect the security of others. In addition, if the key used to protect a tunnel is only shared by the NVEs on the both sides, the egress NVE receiving a data packet is able to distinctively prove the identity of the ingress NVE encapsulating the data packet during the message authentication.

REQ 14: If there are multicast packets, the security solution of NVO3 SHOULD be able to assign distinct cryptographic group keys to protect the multicast packets exchanged among the NVEs within different multicast groups.

In practice, a NVE may need to use the multicast capability provided by the underlying network to transfer multicast packets to other NVEs. In this case, in order to provide an essential packet level security protection specified in requirements 11 and 12, at least a group key may need to be shared among the NVEs in a same mutlicast group, in order to provide packet level authentication or optionally confidentiality protection for the multicast packets transferred within the group. It is recommended to deploy different keys for different mutlicast groups, in order to confine the insider attacks on NVEs.

REQ 15: Upon receiving a data packet, an egress NVE must be able to verify whether the packet is from a proper ingress NVE which is authorized to forward that packet.

In cooperation with authentication, authorization enables a egress NVE to detect the data packets which violate certain security policies, even when they are forwarded from a legal NVE. For instance, if a data packet belonging to a VN is forwarded from an ingress NVE which is not supposed to support that VN, the packet needs to be detected and discarded. Note that the detection of a invalid packet may not indicate that the system is under a malicious attack. Mis-configuration or byzantine failure of a NVE may also result in such invalid packets.

## 5.2. Control/Data Plane between NVEs and Hypervisors

Apart from data traffics, the NVE and hypervisors may also need to exchange signaling packets in order to facilitate, e.g., VM online detection, VM migration detection, or auto-provisioning/service discovery [I-D.ietf-nvo3-framework].

A NVE and the hypervisors working with it can be deployed in a distributed way (e.g., the NVE is implemented in an individual

device, and the hypervisors are located on servers) or in a co-located way (e.g., the NVE and the hypervisors are located on the same server). In the former case, the data and control traffic between the NVE and the hypervisors are exchanged over network.

#### 5.2.1. Distributed Deployment of NVE and Hypervisor

Five security requirements applicable for both control and data packets exchanged between NVEs and hypervisors are listed as follows:

REQ 16: The security solution for NVO3 SHOULD enable the communicating NVE and hypervisor to mutually authenticate each other before exchanging any control/ data packets.

Mutual authentication is used to prevent an attacker from impersonating a legal NVE or a hypervisor without being detected and then reduce the risks of man-in-the-middle attacks. A successful authentication normally results in the distribution key materials to protect the security of subsequent communications.

REQ 17: The security solution of NVO3 MUST be able to provide integrity protection, replay protection and origin authentication for the control/ data packets exchanged between a NVE and a hypervisor.

Packet level security protection can prevent an attacker from illegally interfere with the normal operations of NVEs and hypervisors by injecting bogus control packets into the network. In addition, because it is assumed the network connecting the NVE and the hypervisor is potentially accessible to attackers, security solutions need to prevent an attacker locating in the middle between the NVE and the hypervisor from modifying the VN identification information in the packet headers so as to manipulate the NVE to transport the data packets within a VN to another.

REQ 18: If a NVE needs to communicate with multiple hypervisors, the security solution of a NVO3 network SHOULD be able to provide different keys and ciphers to secure the control /data packets exchanged between different hypervisors and their NVEs respectively.

This requirement is used to benefit the damage confinement of inside attacks. For instance, the compromise of a hypervisor will not affect the security of control/data traffics exchanged between the NVE and other hypervisors.

REQ 19: Before accepting a control/data packet, a NVE or a hypervisor receiving the packet MUST verify that the device sending the packet is authorized to do so.

This is an authorization requirement. When receiving a control/data packet, besides authentication, authorization needs to be carried out by a NVE or a hypervisor to identify the role that the packet sender acts as and then assess the sender's privileges. Therefore, if a compromised hypervisor attempts to use its credentials to impersonate a NVE to communicate with other hypervisors, it will be detected.

REQ 20: The security solution of a NVO3 network SHOULD be able to provide different security levels of protections for the control/data traffics exchanged between a NVE or a hypervisor.

The control and data traffics between a NVE and a hypervisor may be transported over the same path or even within the same security channel. However, when the control traffics and data traffics have different levels of sensitivity, the protection on them needs to be different. In this case, the security solution may need to use different security channels for control and data traffics respectively and so protect the data and control traffics exchanged between a hypervisor and a NVE with different keys and ciphers.

#### 5.2.1.1. Control Plane

REQ 21: The security solution of a NVO3 network MAY provide confidentiality protection for the control traffics exchanged between a NVE and a hypervisor.

The contents of the control/data packets need to be encrypted when they are considered to be sensitive.

Similar to REQs 6 and 7, the following two requirements are used to mitigate potential DDoS risks.

REQ 22: The frequency in forwarding control packets from a NVE or a hypervisors MUST be limited.

This is a common security requirement that can effectively avoid the capability of a device in processing control packets to be overwhelmed by the high frequent control packets generated by the devices attached to it.

REQ 23: Amplification effect SHOULD be Addressed.

If the responses generated by a NVE or a hypervisor are much longer than the received requests, an attacker may take advantage of the device as a reflector to perform DDoS attacks. Specifically, the attacker sends a large amount of spoofed short requests to NVEs or hypervisors with the source address of a victim. The responses will then be generated by the NVEs and forwarded to the victim and overwhelm its process capability. This issues should be considered in the design of the control protocols.

#### 5.2.1.2. Data Plane

REQ 24: The security solution of a NVO3 network MUST provide security gateways to control the data traffics across the boundaries of different VNs according to specified security policies.

In [I-D.ietf-nvo3-overlay-problem-statement], the data plane isolation requirement amongst different VNs has been discussed. The traffic within a virtual network can only be transited into another one in a controlled fashion (e.g., via a configured router and/or a security gateway).

REQ 25: The security solution of a NVO3 network MAY provide confidentiality protection for the data traffics exchanged between a NVE and a hypervisor.

When the contents of the data packets are sensitive to a tenant, the data packet needs to be encrypted. The security solution of a NVE network may need to provide confidentiality for the data packets exchanged between a NVE and a hypervisor if they have to use an insecure network to transport their data packet and the tenants cannot encrypt their sensitive data themselves.

### 6. Candidate Techniques

This section introduces the techniques which can potentially be used to fulfill the security requirements introduced in Section 5.

#### 6.1. Entity Authentication

Entity authentication is normally performed as a part of automated key management, and a successful authentication may result in the key materials used in subsequent communications.

The widely adopted protocols supporting entity authentication include: IKE[RFC2409], IKEv2[RFC4306], EAP[RFC4137], TLS [RFC5246] and etc.



It is recommended to cryptographically verify the devices' identities during authentication. Therefore, an inside attacker cannot use the keys or credentials got from the compromised device to impersonate other victims.

## 6.2. Packet Level Security

There are requirements about protecting the integrity, confidentiality, and provide packet origin authentication for control / data packets. Such functions can be provided through using the underlying security protocols (e.g., IPsec AH[RFC4302], IPsec ESP[RFC4303], TLS[RFC5246]). Also, when designing the control protocols people can select to provide embedded security approaches (just like the packet level security mechanism provided in OSPFv2[RFC2328]). The cryptographic keys can be manually deployed or dynamically generated by using certain automatic key management protocols. Note that when using manual key management, the replay protection mechanism of IPsec will be switched off.

## 6.3. Authorization

Without any cryptographic supports, the authorization mechanisms (e.g., packet filters) could be much easier to be bypassed by attackers, and thus the authorization mechanisms deployed on NOV3 devices should interoperate with entity authentication and other packet level security mechanisms, and be able to make the access control decisions based on the cryptographically proved results. An exception is packet filtering. Because packet filters are efficient and can effectively drop some un-authorized packets before they have to be cryptographically verified, it is worthwhile to use packet filters as an auxiliary approach to dealing with some simple attacks and increasing the difficulties of DoS/DDoS attacks targeting at the security protocol implementations.

## 7. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

## 8. Security Considerations

### 8.1. Automated Key Management in NVO3

Because entity authentication and automated key distribution are normally performed in the same process, the requirements of entity authentication have already implied that it is recommended to use

automated key management in the security solutions for NVO3 networks. In the cases where there are a large amount of NVEs working within a NVO3 overlay, manual key management becomes infeasible. First, it could be tedious to deploy pre-shared keys for thousands of NVEs, not to mention that multiple keys may need to be deployed on a single device for different purposes. Key derivation can be used to mitigate this problem. Using key derivation functions, multiple keys for different usages can be derived from a pre-shared master key. However, key derivation cannot protect against the situation where a system was incorrectly trusted to have the key used to perform the derivation. If the master key were somehow compromised, all the resulting keys would need to be changed [RFC4301]. Moreover, some security protocols need the support of automated key management in order to perform certain security functions properly. As mentioned above, the replay protecting mechanism of IPsec will be turned off without the support of automated key management mechanisms.

## 8.2. Issues not Discussed

Because this memo only tries to provide the most essential high level requirements, some important issues in designing concret security mechanisms are not covered in the requirements. Such issues include:

- o How to manage keys/credentials during their life periods
- o How to support algorithm agility
- o How to provide accountability
- o How to secure the management interfaces
- o Use underlying security protocols versus design integrated security extensions

## 9. Acknowledgements

Thanks a lot for the comments from Melinda Shore and Zu Qiang.

## 10. References

### 10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

## 10.2. Informative References

- [I-D.ietf-ipsecme-ad-vpn-problem]  
Manral, V. and S. Hanna, "Auto Discovery VPN Problem Statement and Requirements", draft-ietf-ipsecme-ad-vpn-problem-09 (work in progress), July 2013.
- [I-D.ietf-nvo3-framework]  
Lasserre, M., Balus, F., Morin, T., Bitar, N., and Y. Rekhter, "Framework for DC Network Virtualization", draft-ietf-nvo3-framework-04 (work in progress), November 2013.
- [I-D.ietf-nvo3-overlay-problem-statement]  
Narten, T., Gray, E., Black, D., Fang, L., Kreeger, L., and M. Napierala, "Problem Statement: Overlays for Network Virtualization", draft-ietf-nvo3-overlay-problem-statement-04 (work in progress), July 2013.
- [I-D.kreeger-nvo3-hypervisor-nve-cp]  
Kreeger, L., Narten, T., and D. Black, "Network Virtualization Hypervisor-to-NVE Overlay Control Protocol Requirements", draft-kreeger-nvo3-hypervisor-nve-cp-01 (work in progress), February 2013.
- [I-D.mahalingam-dutt-dcops-vxlan]  
Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", draft-mahalingam-dutt-dcops-vxlan-06 (work in progress), November 2013.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.
- [RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.
- [RFC4046] Baugher, M., Canetti, R., Dondeti, L., and F. Lindholm, "Multicast Security (MSEC) Group Key Management Architecture", RFC 4046, April 2005.
- [RFC4137] Vollbrecht, J., Eronen, P., Petroni, N., and Y. Ohba, "State Machines for Extensible Authentication Protocol (EAP) Peer and Authenticator", RFC 4137, August 2005.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

- [RFC4302] Kent, S., "IP Authentication Header", RFC 4302, December 2005.
- [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.
- [RFC4306] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, December 2005.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.
- [RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

## Authors' Addresses

Sam Hartman  
Painless Security  
356 Abbott Street  
North Andover, MA 01845  
USA

Email: [hartmans@painless-security.com](mailto:hartmans@painless-security.com)  
URI: <http://www.painless-security.com>

Dacheng Zhang  
Huawei  
Beijing  
China

Email: [zhangdacheng@huawei.com](mailto:zhangdacheng@huawei.com)

Margaret Wasserman  
Painless Security  
356 Abbott Street  
North Andover, MA 01845  
USA

Phone: +1 781 405 7464  
Email: [mrw@painless-security.com](mailto:mrw@painless-security.com)  
URI: <http://www.painless-security.com>

NVO3  
Internet-Draft  
Intended status: Standards Track  
Expires: August 16, 2014

P. Jain  
K. Singh  
F. Balus  
Nuage Networks  
W. Henderickx  
Alcatel-Lucent  
V. Bannai  
PayPal  
R. Shekhar  
A. Lohiya  
Juniper Networks  
February 12, 2014

Generic Overlay OAM and Datapath Failure Detection  
draft-jain-nvo3-overlay-oam-01

Abstract

This proposal describes a mechanism that can be used to detect Data Path Failures of various overlay technologies as VXLAN, NVGRE, MPLSoGRE and MPLSoUDP and verifying/sanity of their Control and Data Plane for given Overlay Segment. This document defines the following for each of the above Overlay Technologies:

- o Encapsulation of OAM Packet, such that it has same Outer and Overlay Header as any End-System's data going over the same Overlay Segment.
- o The mechanism to trace the Underlay that is exercised by any Overlay Segment.
- o Procedure to verify presence of any given Tenant VM or End-System within a given Overlay Segment at Overlay End-Point.

Even though the present proposal addresses Overlay OAM for VXLAN, NVGRE, MPLSoGRE and MPLSoUDP, but the procedures described are generic enough to accommodate OAM for any other Overlay Technology.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 16, 2014.

#### Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	6
2. Terminology . . . . .	8
3. Motivation for Overlay OAM . . . . .	9
4. Approach . . . . .	10
5. Packet Format . . . . .	11
5.1. Overlay OAM Encapsulation in Layer 2 Context . . . . .	11
5.2. Overlay OAM Encapsulation in Layer 3 Context . . . . .	11
5.3. Generic Overlay OAM Packet Format . . . . .	11
5.3.1. TLV Types for various Overlay Ping Models . . . . .	14
5.3.1.1. TLV for VXLAN Ping . . . . .	15
5.3.1.2. TLV for NVGRE Ping . . . . .	16
5.3.1.3. TLV for MPLSoGRE Ping . . . . .	17
5.3.1.4. TLV for MPLSoUDP Ping . . . . .	18
6. Return Codes . . . . .	19
7. Procedure for Overlay Segment Ping . . . . .	20
7.1. Encoding of Inner Header for Echo Request in Layer 2 Context . . . . .	20
7.2. Encoding of Inner Header for Echo Request in Layer 3 Context . . . . .	21
7.3. VXLAN Procedures . . . . .	21
7.3.1. Sending VXLAN Echo Request . . . . .	21
7.3.2. Receiving VXLAN Echo Request . . . . .	22
7.3.3. Sending VXLAN Echo Reply . . . . .	23
7.3.4. Receiving VXLAN Echo Reply . . . . .	23
7.4. NVGRE Procedures . . . . .	23
7.4.1. Sending NVGRE Echo Request . . . . .	23
7.4.2. Receiving NVGRE Echo Request . . . . .	24
7.4.3. Sending NVGRE Echo Reply . . . . .	25
7.4.4. Receiving NVGRE Echo Reply . . . . .	25
7.5. MPLSoGRE Procedures . . . . .	25
7.5.1. Sending MPLSoGRE Echo Request . . . . .	25
7.5.2. Receiving MPLSoGRE Echo Request . . . . .	25
7.5.3. Sending MPLSoGRE Echo Reply . . . . .	26
7.5.4. Receiving MPLSoGRE Echo Reply . . . . .	26
7.6. MPLSoUDP Procedures . . . . .	26
7.6.1. Sending MPLSoUDP Echo Request . . . . .	26
7.6.2. Receiving MPLSoUDP Echo Request . . . . .	27
7.6.3. Sending MPLSoUDP Echo Reply . . . . .	28
7.6.4. Receiving MPLSoUDP Echo Reply . . . . .	28

8. Procedure for Trace . . . . .	29
9. Procedure for End-System Ping . . . . .	30
9.1. Sub-TLV for End-System Ping . . . . .	30
9.1.1. Sub-TLV for Validating End-System MAC Address . . . . .	31
9.1.2. Sub-TLV for Validating End-System IP Address . . . . .	32
9.1.3. Sub-TLV for Validating End-System MAC and IP Address . . . . .	33
9.2. Sending End-System Ping Request . . . . .	34
9.3. Receiving End-System Ping Request . . . . .	34
9.4. Sending End-System Ping Reply . . . . .	35
9.5. Receiving End-System Ping Reply . . . . .	35
10. Security Considerations . . . . .	37
11. Management Considerations . . . . .	38
12. Acknowledgements . . . . .	39
13. IANA Considerations . . . . .	40
14. References . . . . .	41
14.1. Normative References . . . . .	41
14.2. Informative References . . . . .	42
Authors' Addresses . . . . .	43



The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119 [RFC2119].

When used in lower case, these words convey their typical use in common language, and are not to be interpreted as described in RFC2119 [RFC2119].

## 1. Introduction

VXLAN [I-D.draft-mahalingam-dutt-dcops-vxlan], NVGRE [I-D.draft-sridharan-virtualization-nvgre], MPLSoGRE [RFC4023] and MPLSoUDP [I-D.draft-ietf-mpls-in-udp] are well known technologies and are used as tunneling mechanism to Overlay either Layer 2 networks or Layer 3 networks on top of Layer 3 Underlay networks. For all above Overlay Models there are two Tunnel End Points for a given Overlay Segment. One End Point is where the Overlay Originates, and other where Overlay Terminates. In most cases the Tunnel End Point is intended to be at the edge of the network, typically connecting an access switch to an IP transport network. The access switch could be a physical or a virtual switch located within the hypervisor on the server which is connected to End System which is a VM.

This document describes a mechanism that can be used to detect Data Plane failures and sanity of Overlay Control and Data Plane for a given Overlay Segment, and the method to trace the Underlay path that is exercised by any given Overlay Segment.

The document also defines procedures for validating the presence of any given Tenant VM/End-System/End-System or Flow representing the End-System System within a given Overlay Segment.

The proposal describes:

- o The mechanism to verify Overlay Control Plane and Data Plane consistency at the Overlay End Point(s), by encapsulating the OAM Packet in exact the same way as that of any End System Traffic that is transported over the Overlay Segment.
- o The mechanism to trace the Underlay that is exercised by any Overlay Segment.
- o The mechanism to verify presence of any "End-System" in a given Overlay Segment.

The proposal defines the information to check correct operation of the Data Plane, as well as a mechanism to verify the Data Plane against the Control Plane for a given Overlay Segment.

It is important consideration in this proposal to carry Echo Request along same Data Path that any End System's data using the given Overlay Segment takes.

The tenants VM(s) or End System(s) are not aware of the Overlays and as such the need for the verification of the Data Path MUST solely rest with the Cloud Provider. The use cases where the Tenant VM(s)

need to be aware of the Data Plane failures is beyond the scope of this document.

## 2. Terminology

Terminology used in this document:

OAM: Operations, Administration, and Management

VXLAN: Virtual eXtensible Local Area Network.

NVGRE: Network Virtualization using GRE.

MPLSoGRE: Encapsulating MPLS in IP or Generic Routing Encapsulation (GRE)

MPLSoUDP: Encapsulating MPLS in UDP.

Originating End Point: Overlay Segment's Head End or Starting Point of Overlay Tunnel.

Terminating End Point: Overlay Segment's Tail End or Terminating Point of Overlay Tunnel.

VM: Virtual Machine.

VNI: VXLAN Network Identifier (or VXLAN Segment ID)

VSID: Virtual Subnet ID. (for NVGRE)

NVE: Network Virtualized Edge

End System: Could be Tenant VM, Host, Bridge etc. - System whose data is expected to go over Overlay Segment.

Echo Request: Throughout this document, Echo Request packet is expected to be transmitted by Originator Overlay End Point and destined to Overlay Terminating End Point.

Echo Reply: Throughout this document, Echo Reply packet is expected to be transmitted by Terminating Overlay End Point and destined to Overlay Originating End Point.

Other terminologies are as defined in  
[I-D.draft-mahalingam-dutt-dcops-vxlan],  
[I-D.draft-sridharan-virtualization-nvgre], [RFC4023] and  
[I-D.draft-ietf-mpls-in-udp]

### 3. Motivation for Overlay OAM

When any Overlay Segment fails to deliver user traffic, there is a need to provide a tool that would enable users, as Cloud Providers to detect such failures, and a mechanism to isolate faults. It may also be desirable to test the data path before mapping End System traffic to the Overlay Segment.

The basic idea is to facilitate following verifications:-

- o End-System's data that are expected to go over a particular Overlay Segment actually ends up using the Data-Path represented by given Overlay Segment between the two End-Points.
- o To verify the correct value of Overlay Segment Identifier is programmed at Originating and Terminating End Point(s) for a given Overlay Segment. Segment Identifier will be VNI for VXLAN, VSID for NVGRE, MPLS Label for MPLSoGRE and MPLSoUDP.
- o The facilitate mechanism to trace the Underlay that is exercised by any Overlay Segment.
- o The mechanism to verify presence of any "End-System" in a given Overlay Segment.

To facilitate verification of Overlay Segment or any End-System using the Overlay, this document proposes sending of a Packet (called an "Echo Request") along the same data path as other Packets belonging to this Segment. Echo Request also carries information about the Overlay Segment whose Data Path is to be verified. This Echo Request is forwarded just like any other End System Data Packet belonging to that Overlay Segment, as it contains the same Overlay Encapsulation as regular End System's data.

On receiving Echo Request at the end of the Overlay Segment, it is sent to the Control Plane of the Terminating Overlay End Point, which in-turn would respond with Echo Reply.

To facilitate tracing of the Underlay used by any given Overlay Segment, the document proposes Echo Request/Reply encapsulation in "trace mode", which would allow the user or Cloud Provider to gather information of the Underlay network.

#### 4. Approach

The proposal aims at validating Data Plane and its view of Control Plane for a particular Overlay Segment. To achieve this aim, the draft proposes creating an Overlay OAM Packet which MUST be encapsulated with the Overlay Header as that of any End-Point data going over the same Overlay Segment. This would guarantee the data-path for OAM Packet follows the same path as that for any End User data going over the same Overlay Segment.

The draft outlines procedures to encode Overlay Header and Inner Ethernet or IP Header based on the type of payload that Overlay is expected to carry.

## 5. Packet Format

Generic Overlay Echo Request/Reply is a UDP Packet identified by well known UDP Port XXXX. The payload carried by Overlay typically could be either be Layer 2 / Ethernet Frame, or it could be Layer 3 / IP Packet.

### 5.1. Overlay OAM Encapsulation in Layer 2 Context

If the encapsulated payload carried by Overlay is of type Ethernet, then the OAM Echo Request packet would have inner Ethernet Header, followed by IP and UDP Header. The payload of inner UDP would be as described in below section "Generic Overlay OAM Packet Format".

### 5.2. Overlay OAM Encapsulation in Layer 3 Context

If the encapsulated payload carried by Overlay is of type IP, then the OAM Echo Request packet would have inner IP Header, followed by UDP Header. The payload of inner UDP would be as described in below section "Generic Overlay OAM Packet Format".

### 5.3. Generic Overlay OAM Packet Format

Following is the format of UDP payload of Generic Overlay OAM Packet:

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
| Message Type | Reply mode | Return Code | Return Subcode|
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Originator Handle                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Sequence Number                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     TimeStamp Sent (seconds)                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     TimeStamp Sent (microseconds)                       |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     TimeStamp Received (seconds)                       |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     TimeStamp Received (microseconds)                   |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     TLVs ...                                           |
.                                                                                   .
.                                                                                   .
.                                                                                   .
+-----+-----+-----+-----+-----+-----+-----+-----+

```

#### Generic Overlay OAM Packet

The Message Type is one of the following:-

Value	What it means
1	Echo Request
2	Echo Reply

Reply Mode Values:-

Value	What it means
1	Do not reply
2	Reply via an IPv4/IPv6 UDP Packet
3	Reply via Overlay Segment

Echo Request with 1 (Do not reply) in the Reply Mode field may be used for one-way connectivity tests. The receiving node may log gaps in the Sequence Numbers and/or maintain delay/jitter statistics. For normal operation Echo Request would have 2 (Reply via an IPv4 UDP



Packet) in the Reply Mode field.

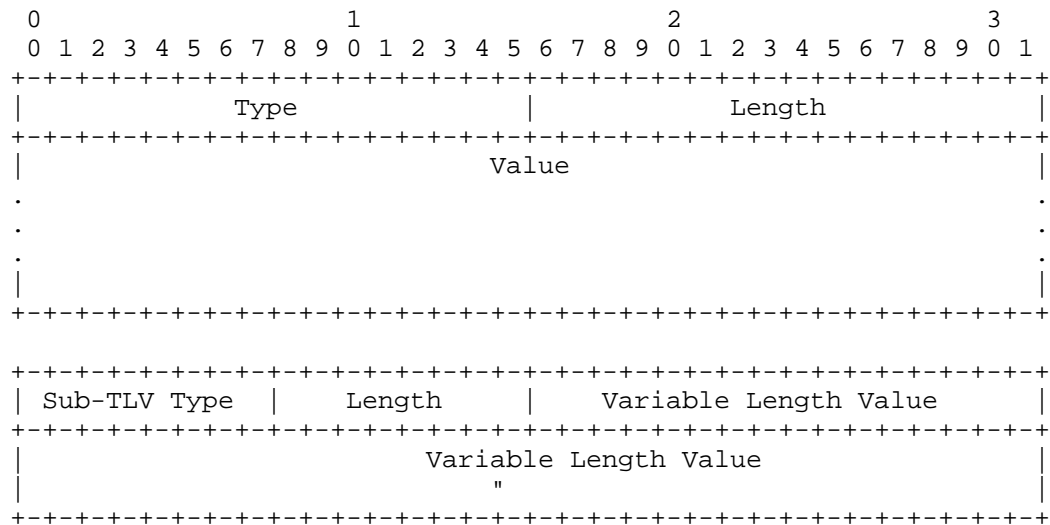
If it is desired that the reply also comes back via Overlay Segment i.e. encapsulated with the Overlay Header, then the Reply Mode field needs to be set to 3 (Reply via Overlay Segment).

The Originator's Handle is filled in by the Originator, and returned unchanged by the receiver in the Echo Reply (if any). The value used for this field can be implementation dependent, this MAY be used by the Originator for matching up requests with replies.

The Sequence Number is assigned by the Originator of Echo Request and can be (for example) used to detect missed replies.

The TimeStamp Sent is the time-of-day (in seconds and microseconds, according to the sender's clock) in NTP format [NTP] when the VXLAN Echo Request is sent. The TimeStamp Received in an Echo Reply is the time-of-day (according to the receiver's clock) in NTP format that the corresponding Echo Request was received.

TLVs (Type-Length-Value tuples) have the following format:



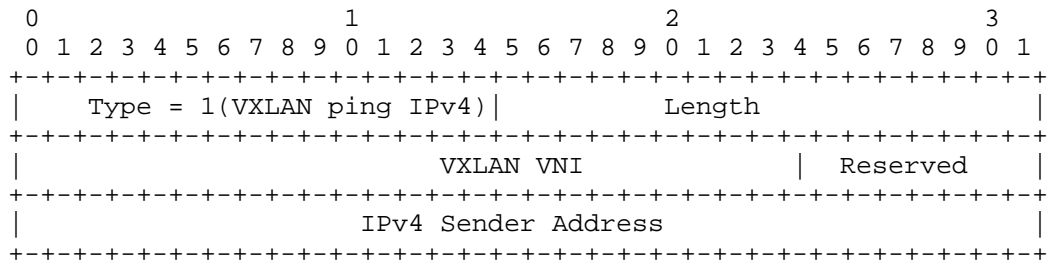
Types are defined below; Length is the length of the Value field in octets. The Value field depends on the Type; it is zero padded to align to a 4-octet boundary. There could be one or many optional Sub-TLV that could be encoded under the TLV.

## 5.3.1. TLV Types for various Overlay Ping Models

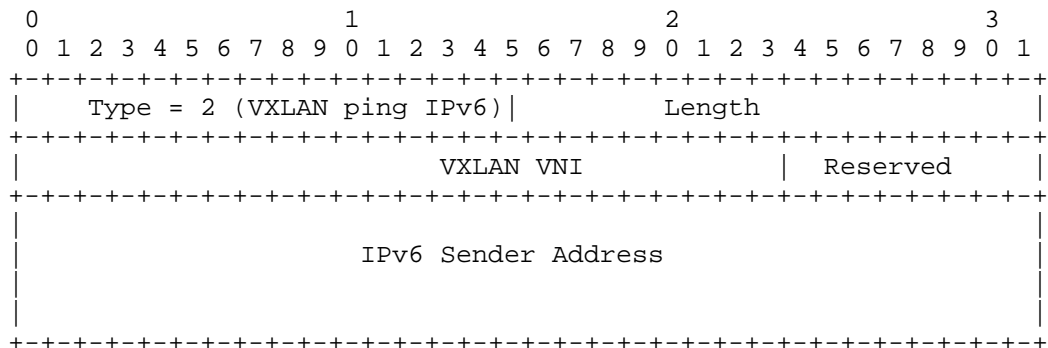
## TLV Types:-

Value	What it means
1	VXLAN Segment Ping for IPv4
2	VXLAN Segment Ping for IPv6
3	NVGRE Segment Ping for IPv4
4	NVGRE Segment Ping for IPv6
5	MPLSoGRE Segment Ping for IPv4
6	MPLSoGRE Segment Ping for IPv6
7	MPLSoUDP Segment Ping for IPv4
8	MPLSoUDP Segment Ping for IPv6

## 5.3.1.1. TLV for VXLAN Ping

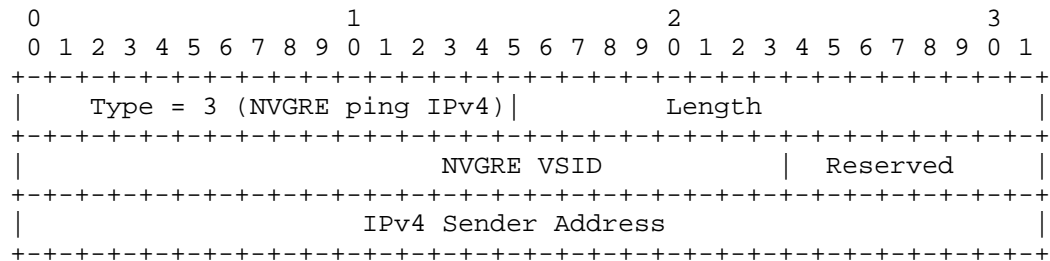


TLV if Sender Address is IPv4

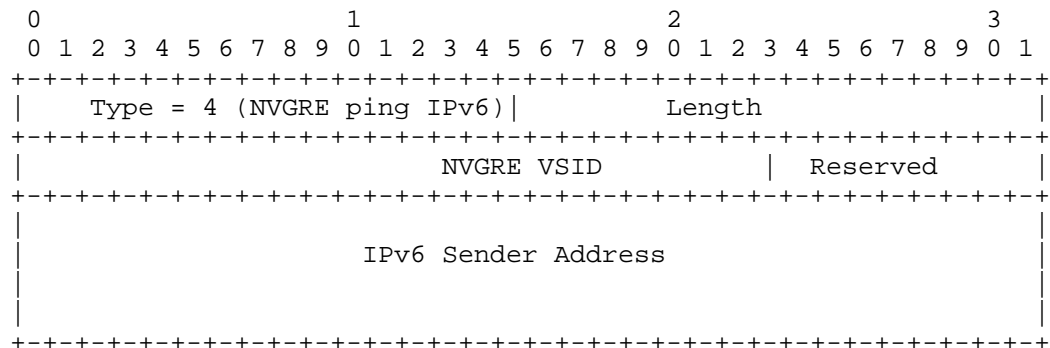


TLV if Sender Address is IPv6

## 5.3.1.2. TLV for NVGRE Ping

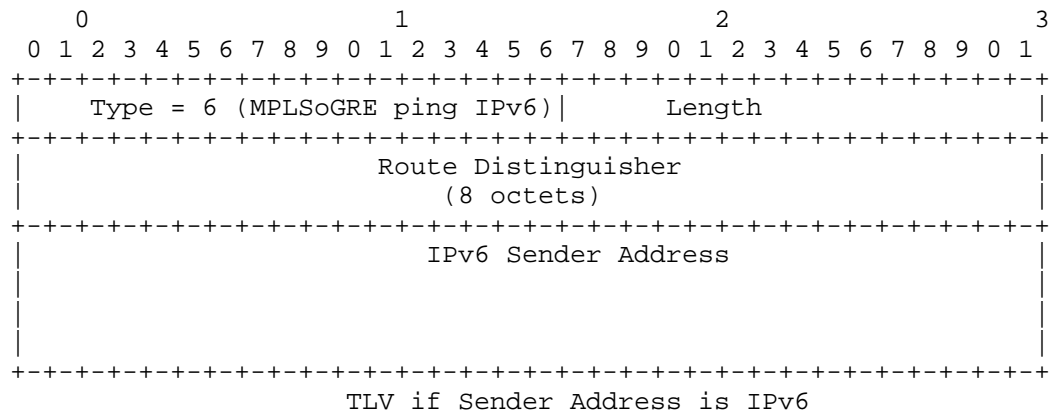
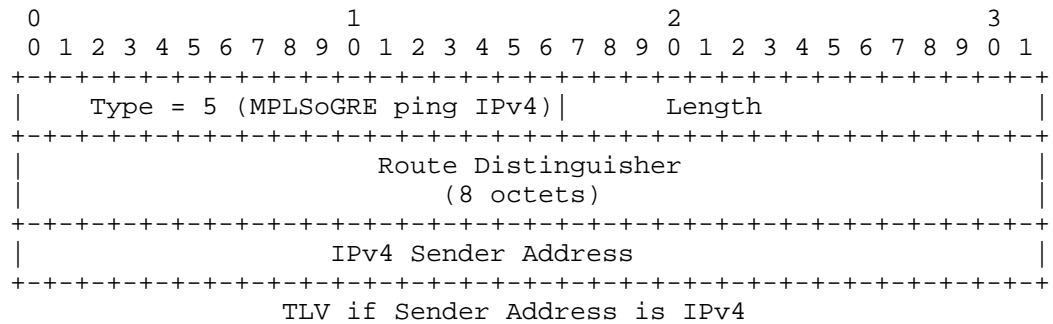


TLV if Sender Address is IPv4



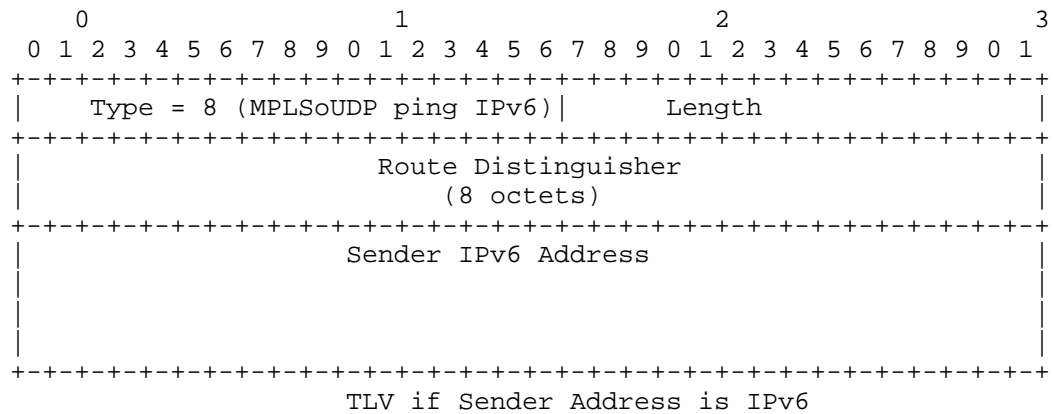
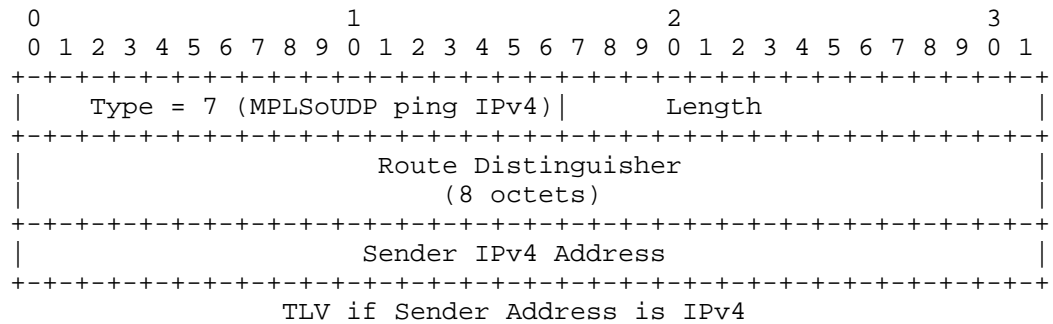
TLV if Sender Address is IPv6

## 5.3.1.3. TLV for MPLSoGRE Ping



Route Distinguisher is defined as part of [RFC4365]

## 5.3.1.4. TLV for MPLSoUDP Ping



Route Distinguisher is defined as part of [RFC4365]

## 6. Return Codes

Sender MUST always set the Return Code set to zero. The receiver can set it to one of the values listed below when replying back to Echo-Request.

Following are the Return Codes (Suggested):-

Value	What it means
-----	-----
0	No return code
1	Malformed Echo Request Received
2	Overlay Segment Not Present
3	Overlay Segment Not Operational
4	Return-Code-OK

## 7. Procedure for Overlay Segment Ping

Echo Request is used to test Data Plane and its view of Control Plane for particular Overlay Segment. The Overlay Segment to be verified is identified differently for various Overlay Technologies. For VXLAN, VNI is used to identify given Overlay Segment. For NVGRE, VSID is used. For MPLSoGRE and MPLSoUDP the MPLS Stack is used to identify a given Overlay Segment.

For the Data Plane verification, the Overlay Echo Request Packet MUST be encapsulated within the Overlay Header, which is same as that of any End-Point data going over the same Overlay Segment. This would guarantee the data-path for OAM Packet follows the same path as that for any End User data going over the same Overlay Segment.

The payload carried by Overlay typically could be either be Layer 2 or Ethernet Frame, or it could be Layer 3 or IP Packet. Based on the type of payload following is the way inner Header(s) of Echo Request would be encoded.

### 7.1. Encoding of Inner Header for Echo Request in Layer 2 Context

If the encapsulated payload carried by Overlay is of type Ethernet, then the OAM Echo Request packet would have inner Ethernet Header, followed by IP and UDP Header. The payload of inner UDP would be as described in below section "Generic Overlay OAM Packet Format".

Inner Ethernet Header for the Echo Request Packet MUST have the Destination Mac set to 00-00-5E-90-XX-XX (to be assigned IANA). The Source Mac should be set to Mac Address of the Originating VTEP. However, it is desired that the Inner Source Mac SHOULD not be learnt in the MAC-Table as this represent Control Packet in context of Overlay OAM.

Inner IP header is set with the Source IP Address which is a routable Address of the sender; the Destination IP Address is a (randomly chosen) IPv4 Address from the range 127/8, IPv6 addresses are chosen from the range 0:0:0:0:0:FFFF:127/104. The IP TTL is set to 255.

The inner Destination UDP port is set to xxxx (assigned by IANA for Overlay OAM).

The "Generic Overlay OAM Packet" will now be encoded, with following information.

The sender chooses a Originator's Handle and a Sequence Number. When sending subsequent Overlay Echo Requests, the sender SHOULD increment the Sequence Number by 1.



The TimeStamp Sent is set to the time-of-day (in seconds and microseconds) that the Echo Request is sent. The TimeStamp Received is set to zero. Also, the Reply Mode must be set to the desired reply mode. The Return Code and Subcode are set to zero.

Next, the TLV is Encoded for desired Overlay Type, as per Section "Types of TLVs defined for various Overlay Ping Models"

## 7.2. Encoding of Inner Header for Echo Request in Layer 3 Context

If the encapsulated payload carried by Overlay is of type IP, then the Encoding of the Echo Request would be same as above Section "Encoding of Inner Header for Echo Request in Layer 2 Context", but without the presence of Inner Ethernet Header.

## 7.3. VXLAN Procedures

### 7.3.1. Sending VXLAN Echo Request

The Outer VxLAN header for the Echo Request packet follows the encapsulation as defined in [I-D.draft-mahalingam-dutt-dcops-vxlan]. The VNI is same as that of the VXLAN Segment that is being verified. This would make sure that OAM Packet takes the same datapath as any other End System data going over this VXLAN Segment.

The VXLAN Router Alert option

[I-D.draft-singh-nvo3-vxlan-router-alert] MUST be set in the VXLAN header as shown below.

VXLAN Header:

```

+++++
|R|R|R|R|I|R|R|RA|          Reserved          |
+++++
|          VXLAN Network Identifier (VNI) |   Reserved   |
+++++

```

RA: Router Alter Bit (Proposed)

Originating VTEP MAY set the I Bit to 0 in VXLAN Header when sending OAM Frame. This would cause dropping of such VXLAN frames on any Terminating VTEP that does not understand Overlay OAM framework, and prevent sending those frames to End-Systems or VMs.

It is desired to choose the Source UDP port (in the outer header), so as to exercise the same Data-Path as that of the traffic carried over the VXLAN Segment and is left to the implementation.

The Encoding of Inner Header(s) and UDP payload of Generic Overlay OAM Packet is as described in above Sub-Section i.e. "Encoding of Inner Header for Echo Request in Layer 2/Layer 3 Context".

### 7.3.2. Receiving VXLAN Echo Request

At the Terminating Overlay End Point or VTEP, since the Overlay OAM Packet is exactly same as that of End-System Packet(s). It is important to send OAM packet to Control Plane and prevent it from sending to the End System. The trapping and sending VXLAN Echo Request to the Control Plane is triggered by one of the following Packet processing exceptions: VXLAN Router Alert option, [I-D.draft-singh-nvo3-vxlan-router-alert] the Inner Destination MAC Address of 00-00-5E-90-XX-XX as defined in above section, and the Destination IP Address in the 127/8 Address range for IPv4 Address, or 0:0:0:0:0:FFFF:127/104 for IPv6 Address.

The Control Plane further identifies the Overlay OAM Application by UDP well know destination port xxxx.

Since the VxLAN Router Alert bit is set in VxLAN Header, which signifies the presence of Control Packet. The terminating VTEP SHOULD not learn the Mac address set in the Inner Mac Header of VxLAN Echo Request Packet.

Once the VXLAN Echo Request Packet is identified at Control Plane, it is processed as follows:-

- o General Packet sanity is verified. If the Packet is not well-formed, VTEP SHOULD send VXLAN Echo Reply with the Return Code set to "Malformed Echo Request received" and the Subcode to zero. The header fields Originator's Handle, Sequence Number, and Timestamp Sent are not examined, but are included in the VXLAN Echo Reply message
- o VNI Validation: If there is no entry for VNI, it indicates that there could be a transient or permanent disconnect between Control Plane and data Plane and VTEP needs to report an error with Return Code of "Overlay Segment Not Present" and a Return Subcode of Zero. If the mapping for VNI Exists, but the state is not Operational, VTEP needs to report an error with Return Code of "Overlay Segment Not Operational" If the mapping exists then send VXLAN Echo Reply with a Return Code of "Return-Code-OK", and a Return Subcode of Zero. The procedures for sending the Echo Reply are found in subsection below section.

### 7.3.3. Sending VXLAN Echo Reply

If the Reply Mode is set to "Reply via an IPv4/IPv6 UDP Packet", the Echo Reply is a UDP Packet. It MUST ONLY be sent in response to Echo Request. The Source IP Address in the Header should be Routable Address of the replier; The Destination IP Address should be IP Address of the Echo Request's Originating End Point or the requester. The destination UDP Port is set to XXXX (assigned by IANA for identifying VXLAN OAM application). The IP TTL is set to 255.

The format of the Echo Reply is the same as the Echo Request. The Originator Handle, the Sequence Number, and TimeStamp Sent are copied from the Echo Request; the TimeStamp Received is set to the time-of-day that the Echo Request is received (note that this information is most useful if the time-of-day clocks on the requester and the replier are synchronized). The replier MUST fill in the Return Code and Subcode, as determined in the previous subsection.

If the Reply Mode is set to "Reply via Overlay Segment", then the Replying Overlay End Point is expected to place Echo Reply packet in-band in the Overlay Segment destined to the Originating Overlay End Point. The detailed encapsulation for this would be covered in next revision of the draft.

### 7.3.4. Receiving VXLAN Echo Reply

An Originating Overlay End Point should only receive Echo Reply in response to an Echo Request that it sent. When the Reply Mode is "Reply via an IPv4/IPv6 UDP Packet", the Echo Reply would be an IP Packet/UDP Packet, and is identified by the destination UDP Port XXXX. The Originating Overlay End Point should parse the Packet to ensure that it is well-formed, then attempt to match up the Echo Reply with an Echo Request that it had previously sent, and the Originator Handle. If no match is found, then it should drop the Echo Reply Packet; otherwise, it checks the Sequence Number to see if it matches.

## 7.4. NVGRE Procedures

### 7.4.1. Sending NVGRE Echo Request

The Outer NVGRE header for the Echo Request packet follows the encapsulation as defined in [I-D.draft-sridharan-virtualization-nvgre]. The VSID is same as that of the NVGRE Segment that is being verified. This would make sure that OAM Packet takes the same datapath as any other End System data going over this NVGRE Segment.

The NVGRE Router Alert option

[I-D.draft-singh-nvo3-nvgre-router-alert] MUST be set in the NVGRE header as shown below.

GRE Header:

```

+-----+
|0| |1|0| Reserved0      RA| Ver |   Protocol Type 0x6558   |
+-----+
|                               Virtual Subnet ID (VSID)           | Reserved |
+-----+

```

RA: Router Alter Bit (Proposed)

The Encoding of Inner Header(s) and UDP payload of Generic Overlay OAM Packet is as described in above Sub-Section i.e. "Encoding of Inner Header for Echo Request in Layer 2/Layer 3 Context".

#### 7.4.2. Receiving NVGRE Echo Request

At the Terminating Overlay End Point, since the Overlay OAM Packet is exactly same as that of End-System Packet(s). It is important to send OAM packet to Control Plane and prevent it from sending to the End System. The trapping and sending NVGRE Echo Request to the Control Plane is triggered by one of the following Packet processing exceptions: NVGRE Router Alert option, [I-D.draft-singh-nvo3-nvgre-router-alert] the Inner Destination MAC Address of 00-00-5E-90-XX-XX as defined in above section, and the Destination IP Address in the 127/8 Address range for IPv4 Address, or 0:0:0:0:0:FFFF:127/104 for IPv6 Address.

The Control Plane further identifies the Overlay OAM Application by UDP well know destination port xxxx.

Since the NVGRE Router Alert bit is set in NVGRE Header, which signifies the presence of Control Packet. The Terminating Overlay End Point SHOULD not learn the Mac address set in the Inner Mac Header of NVGRE Echo Request Packet.

Once the NVGRE Echo Request Packet is identified at Control Plane, it is processed as follows:-

- o General Packet sanity is verified. If the Packet is not well-formed, NVGRE End Point SHOULD send NVGRE Echo Reply with the Return Code set to "Malformed Echo Request received" and the Subcode to zero. The header fields Originator's Handle, Sequence Number, and Timestamp Sent are not examined, but are included in the NVGRE Echo Reply message

- o VSID Validation: If there is no entry for VSID, it indicates that there could be a transient or permanent disconnect between Control Plane and data Plane and NVGRE End Point needs to report an error with Return Code of "Overlay Segment Not Present" and a Return Subcode of Zero. If the mapping for VSID Exists, but the state is not Operational, NVGRE End Point needs to report an error with Return Code of "Overlay Segment Not Operational" If the mapping exists then send NVGRE Echo Reply with a Return Code of "Return-Code-OK", and a Return Subcode of Zero. The procedures for sending the Echo Reply are found in subsection below section.

#### 7.4.3. Sending NVGRE Echo Reply

The procedure for sending NVGRE Echo Reply are exactly same as defined in above section "Sending VXLAN Echo Reply".

#### 7.4.4. Receiving NVGRE Echo Reply

The procedure for Receiving NVGRE Echo Reply are exactly same as defined in above section "Receiving VXLAN Echo Reply".

### 7.5. MPLSoGRE Procedures

#### 7.5.1. Sending MPLSoGRE Echo Request

The Outer header of MPLSoGRE for the Echo Request packet follows the encapsulation as defined in [RFC4023]. The MPLS Stack is same as that of the MPLSoGRE Segment that is being verified. This would make sure that OAM Packet takes the same datapath as any other End System data going over this MPLSoGRE Segment.

However, the bottommost Label in MPLS Stack MUST be MPLS Router Alert Label [RFC3032]. This would indicate the Overlay Terminating End Point that the payload is a Control Packet and needs to be delivered to Control Plane.

The Encoding of Inner Header(s) and UDP payload of Generic Overlay OAM Packet is as described in above Sub-Section i.e. "Encoding of Inner Header for Echo Request in Layer 2/Layer 3 Context".

#### 7.5.2. Receiving MPLSoGRE Echo Request

At the Terminating Overlay End Point, since the Overlay OAM Packet is exactly same as that of End-System Packet(s). It is important to send OAM packet to Control Plane and prevent it from sending to the End System. The trapping and sending MPLSoGRE Echo Request to the Control Plane is triggered by one of the following Packet processing exceptions: MPLS Router Alert Label, and the Destination IP Address

in the 127/8 Address range for IPv4 Address, or 0:0:0:0:0:FFFF:127/104 for IPv6 Address.

The Control Plane further identifies the Overlay OAM Application by UDP well know destination port xxxx.

Once the MPLSoGRE Echo Request Packet is identified at Control Plane, it is processed as follows:-

- o General Packet sanity is verified. If the Packet is not well-formed, MPLSoGRE End Point SHOULD send MPLSoGRE Echo Reply with the Return Code set to "Malformed Echo Request received" and the Subcode to zero. The header fields Originator's Handle, Sequence Number, and Timestamp Sent are not examined, but are included in the MPLSoGRE Echo Reply message
- o Segment Validation: If there is no entry for service represented by given Route Distinguisher for the MPLSoGRE Segment, it indicates that there could be a transient or permanent disconnect between Control Plane and Data Plane and MPLSoGRE End Point needs to report an error with Return Code of "Overlay Segment Not Present" and a Return Subcode of Zero. If the entry for service represented by given Route Distinguisher for the MPLSoGRE Segment is present, but is Operationally Down. The End Point needs to report an error with Return Code of "Overlay Segment Not Operational" If the mapping of service represented by given Route Distinguisher for the MPLSoGRE Segment is present and Active, then send MPLSoGRE Echo Reply with a Return Code of "Return-Code-OK".

#### 7.5.3. Sending MPLSoGRE Echo Reply

The procedure for sending MPLSoGRE Echo Reply are exactly same as defined in above section "Sending VXLAN Echo Reply".

#### 7.5.4. Receiving MPLSoGRE Echo Reply

The procedure for Receiving MPLSoGRE Echo Reply are exactly same as defined in above section "Receiving VXLAN Echo Reply".

### 7.6. MPLSoUDP Procedures

#### 7.6.1. Sending MPLSoUDP Echo Request

The Outer header of MPLSoUDP for the Echo Request packet follows the encapsulation as defined in [I-D.draft-ietf-mpls-in-udp]. The MPLS Stack is same as that of the MPLSoUDP Segment that is being verified. This would make sure that OAM Packet takes the same datapath as any other End System data going over this MPLSoUDP Segment.

However, the bottommost Label in MPLS Stack MUST be MPLS Router Alert Label [RFC3032]. This would indicate the Overlay Terminating End Point that the payload is a Control Packet and needs to be delivered to Control Plane.

It is desired to choose the Source UDP port (in the outer header), so as to exercise the same Data-Path as that of the traffic carried over the MPLSoUDP Segment and is left to the implementation.

The Encoding of Inner Header(s) and UDP payload of Generic Overlay OAM Packet is as described in above Sub-Section i.e. "Encoding of Inner Header for Echo Request in Layer 2/Layer 3 Context".

#### 7.6.2. Receiving MPLSoUDP Echo Request

At the Terminating Overlay End Point, since the Overlay OAM Packet is exactly same as that of End-System Packet(s). It is important to send OAM packet to Control Plane and prevent it from sending to the End System. The trapping and sending MPLSoGRE Echo Request to the Control Plane is triggered by one of the following Packet processing exceptions: MPLS Router Alert Label, and the Destination IP Address in the 127/8 Address range for IPv4 Address, or 0:0:0:0:0:FFFF:127/104 for IPv6 Address.

The Control Plane further identifies the Overlay OAM Application by UDP well know destination port xxxx.

Once the MPLSoUDP Echo Request Packet is identified at Control Plane, it is processed as follows:-

- o General Packet sanity is verified. If the Packet is not well-formed, MPLSoUDP End Point SHOULD send MPLSoUDP Echo Reply with the Return Code set to "Malformed Echo Request received" and the Subcode to zero. The header fields Originator's Handle, Sequence Number, and Timestamp Sent are not examined, but are included in the MPLSoUDP Echo Reply message
- o Segment Validation: If there is no entry for service represented by given Route Distinguisher for the MPLSoUDP Segment, it indicates that there could be a transient or permanent disconnect between Control Plane and data Plane and MPLSoUDP End Point needs to report an error with Return Code of "Overlay Segment Not Present" and a Return Subcode of Zero. If the entry for service represented by given Route Distinguisher for the MPLSoUDP Segment is present, but is Operationally Down. The End Point needs to report an error with Return Code of "Overlay Segment Not Operational" If the mapping of service represented by given Route Distinguisher for the MPLSoUDP Segment is present and Active, then

send MPLSoUDP Echo Reply with a Return Code of "Return-Code-OK".

#### 7.6.3. Sending MPLSoUDP Echo Reply

The procedure for sending MPLSoGRE Echo Reply are exactly same as defined in above section "Sending VXLAN Echo Reply".

#### 7.6.4. Receiving MPLSoUDP Echo Reply

The procedure for Receiving MPLSoGRE Echo Reply are exactly same as defined in above section "Receiving VXLAN Echo Reply".



## 8. Procedure for Trace

In order to be able to trace the Path that a particular flow in the Overlay takes through the Underlay Network, following mechanism can be used - An overlay Echo Request packet is built and sent using the mechanisms described in the Section "Procedure for Overlay Segment Ping" so that the overlay traceroute follows the same path as the data packet for the overlay segment being traced.

The Echo Request packet in the traceroute mode is sent with the initial TTL set to 1 in the Outer IP header and thereafter incremented by 1 in each successive request. At each transit hop where the TTL expires, an exception is created. Because of this exception, the packet gets delivered to the Control Plane. Control plane can further deliver the packet to the OAM application based on the TTL exception and the specific UDP port XXXX in the incoming overlay echo request packet. If the transit node has the IP reachability to the destination IP address in the outer IP header, it sends back an overlay echo reply response otherwise the Overlay Echo Request is discarded by the Overlay OAM module on the transit nodes. If the transit node does not support overlay OAM functionality, it will simply generate a regular ICMP TTL exceeded response. This could result into "false negatives". The originating Overlay node that generated the OAM echo request SHOULD try sending the echo request with TTL=n+1, n+2, ... to probe the nodes further down the path to the terminating overlay End-point.

At the originating node, when the Echo Reply from the transit node corresponding to the traceroute query is received, it can correlate the incoming Echo Reply with the traceroute query by matching on the sequence numbers in the Overlay Echo Request/Reply packets.

Current revision of this draft limits overlay traceroute capability to fault isolation only. A subsequent version of the draft will include mechanisms to trace all possible paths in the underlay that can be used to carry overlay tunnel traffic.

## 9. Procedure for End-System Ping

In typical Overlay deployment scenarios there is a desired to check the presence of any given Tenant VM/End-System or Flow representing the End-System System within a given Overlay Segment. This draft proposes the way to achieve it via End-System Ping.

The End-System can be identified at Overlay End Point by either its IP Address, Ethernet MAC Address or combination of IP/MAC Address.

In that case, it would be important to verify the End-System connectivity by procedure which goes over the Overlay Segment from Originating Overlay End-Point and verifies the presence of the End-System at the Terminating Overlay End-Point.

The scope of End-System Ping is solely with the Cloud Provider which owns control of the Overlay End Point(s). It is expected that the Overlay End Point traps this request and checks the Presence of the End-System via its MAC Address, Route or Flow information and replies back. There SHOULD not be a case where the End-System Ping is delivered to the actual End-Point.

### 9.1. Sub-TLV for End-System Ping

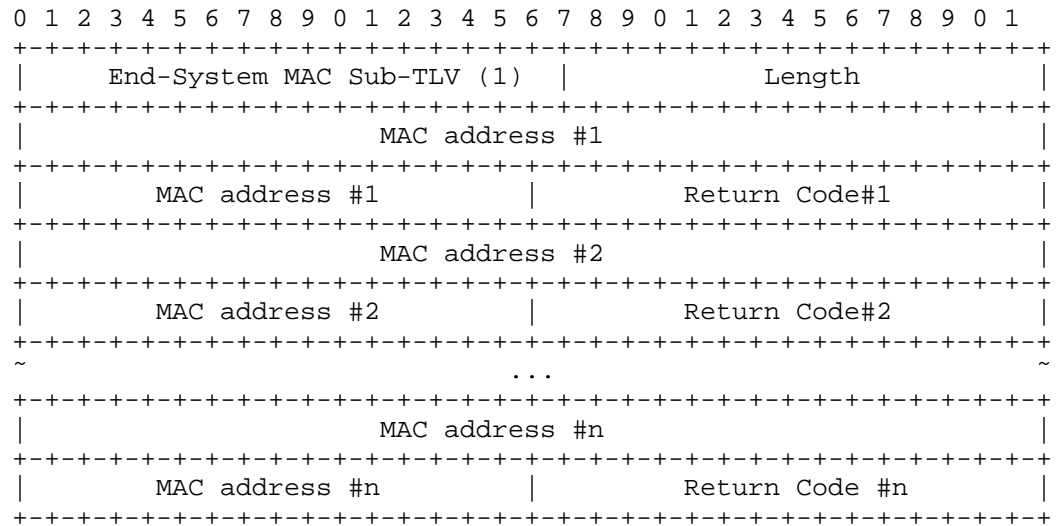
#### Sub-TLV Types:-

Value	What it means
1	End-System MAC Sub-TLV
2	End-System IPv4 Sub-TLV
3	End-System IPv6 Sub-TLV
4	End-System MAC/IPv4 Sub-TLV
6	End-System MAC/IPv6 Sub-TLV

#### End-System Return Code:-

Value	What it means
1	End-System Present
2	End-System Not Present

## 9.1.1.1. Sub-TLV for Validating End-System MAC Address



MAC Address: MAC Address of the End-System, that user is interested to validate.

Return Code: Return Code specifying status of End-System at Overlay End Point

## 9.1.2. Sub-TLV for Validating End-System IP Address

```

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|      End-System IPv4 Sub-TLV (2) |      Length      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     IP address #1      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Return Code #1      |      IP address #2      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      IP address #2      |      Return Code #2      |
+-----+-----+-----+-----+-----+-----+-----+-----+
~                                     ~
...
+-----+-----+-----+-----+-----+-----+-----+-----+
|      IP address #n      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Return Code #n      |
+-----+-----+-----+-----+-----+-----+-----+

```

```

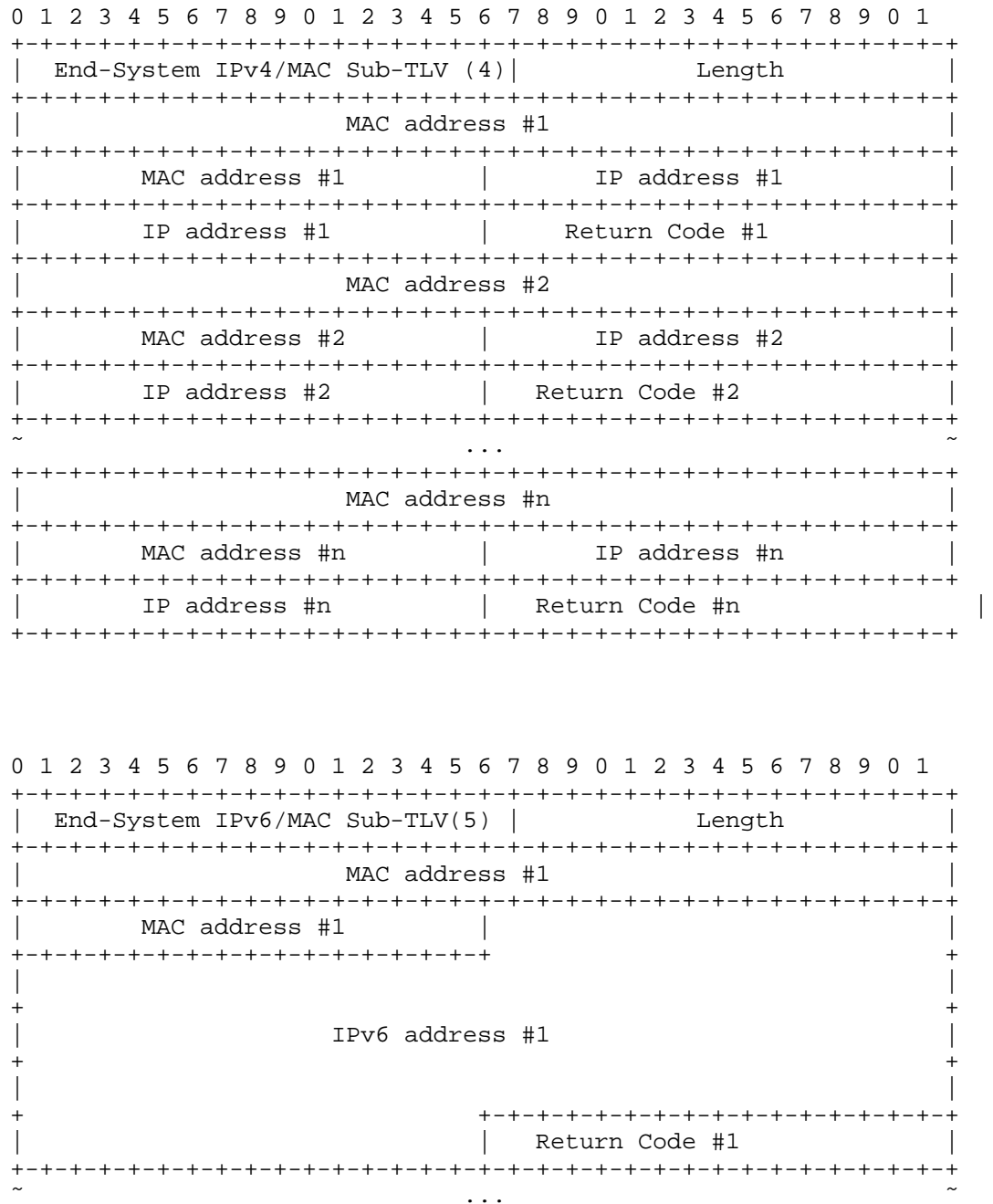
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|      End-System IPv6 Sub-TLV (3) |      Length      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     IPv6 Address #1    |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Return Code #1      |      IPv6 Address #2...  |
+-----+-----+-----+-----+-----+-----+-----+-----+
~                                     ~
...
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     IPv6 Address #n    |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Return Code #n      |
+-----+-----+-----+-----+-----+-----+-----+

```

IP Address : IP Address of the End-System, that user is interested to validate.

Return Code: Return Code specifying status of End-System at Overlay End Point

## 9.1.3. Sub-TLV for Validating End-System MAC and IP Address



```

+-----+
|                                     MAC address #n                                     |
+-----+
|             MAC address #n             |
+-----+
|
+
|             IPv6 address #1             |
+
+
|                                     +-----+
|                                     | Return Code #1 |
|                                     +-----+
+-----+

```

IP Address : IP Address of the End-System, that user is interested to validate.

MAC Address: MAC Address of the End-System, that user is interested to validate.

Return Code: Return Code specifying status of End-System at Overlay End Point

## 9.2. Sending End-System Ping Request

When it is desired to check presence of a given End-System, the Echo Request Message is prepared as described in above Section "Procedure for Overlay Segment Ping". This packet should compose of Outer Header, Overlay Header, Inner Header, Generic Overlay Header with TLV representing desired Overlay Type (VXLAN, NVGRE, MPLSoGRE or MPLSoUDP). Apart from this the packet should also have one of the Sub-TLV's as defined in above section "Sub-TLV for End-System Ping" to identify the type of End-System Ping that user is interested in.

Because of the above mentioned encapsulation, it would be guaranteed that the packet follows the same Data Path as that of any End-User data going over the given Overlay Segment.

User need to fill in MAC, IP or MAC/IP combination for the End-System(s) that needs to be validated at the Overlay End Point in the respective Sub-TLV for End-System Ping.

## 9.3. Receiving End-System Ping Request

On receiving the End-System Ping Request the processing to trap this Packet, and sent it to Control Plane is done by Overlay Terminating End-System as define in above Section "Procedure for Overlay Segment Ping". Once the OAM Packet reaches OAM Application, it is identified as End-System Ping Request by virtue of presence any of the Sub-TLV's as defined in Section "Sub-TLV for End-System Ping".

If the Sub-TLV is of Type "End-System MAC Sub-TLV", the Overlay End Point should iterate through the list of MAC Addresses and verify the presence of individual MAC Address in its Flow Table or MAC Table for the given Overlay Segment.

If the MAC Address is present, it should set the respective End-System's Return Code field in the Sub-TLV to 1 "End-System-Present".

If the MAC Address is not present, it should set respective the End-System's Return Code field in the Sub-TLV to 2 "End-System-Not-Present".

If the Sub-TLV is of Type "End-System IP Sub-TLV", the Overlay End Point should iterate through the list of IP Addresses and verify the presence of individual IP Address in its Flow Table or Route Table for the given Overlay Segment.

If the IP Address is present, it should set the respective End-System's Return Code field in the Sub-TLV to 1 "End-System-Present".

If the IP Address is not present, it should set respective the End-System's Return Code field in the Sub-TLV to 2 "End-System-Not-Present".

If the Sub-TLV is of Type "End-System MAC and IP Sub-TLV", the Overlay End Point should iterate through the list of MAC/IP Addresses and verify the presence of individual MAC/IP Combination in its Flow Table or MAC and IP Table for the given Overlay Segment.

If the IP and MAC Address is present, it should set the respective End-System's Return Code field in the Sub-TLV to 1 "End-System-Present".

If the IP and MAC Address is not present, it should set respective the End-System's Return Code field in the Sub-TLV to 2 "End-System-Not-Present".

#### 9.4. Sending End-System Ping Reply

The procedure for sending End-System Echo Reply is same as defined in above section "Sending VXLAN Echo Reply". The replier MUST fill Sub-TLV with proper Return Code for each element in the End-System Sub-TLV.

#### 9.5. Receiving End-System Ping Reply

An Originating Overlay End Point should only receive Echo Reply for End-System Ping, in response to an Echo Request that it sent. By

virtue of presence of End-System Sub-TLV it would identify the status of respective End-System, and report it to the user. The other part of the handling is similar to section "Receiving VXLAN Echo Reply"



## 10. Security Considerations

TBD

## 11. Management Considerations

None

## 12. Acknowledgements

This document is the outcome of many discussions among many people, including Saurabh Shrivastava, Krishna Ram Kuttuva Jeyaram and Suresh Boddapati of Nuage Networks, Jorge Rabadan of Alcatel-Lucent, Inc and Rahul Kasralikar of Juniper Networks, Inc.

### 13. IANA Considerations

Action-1: This specification reserves a IANA UDP Port Number to be used when sending the Overlay OAM Packet

Action-2: This specification reserves a IANA Ethernet unicast Address for VXLAN/NVGRE Exception handling. This Address needs to be reserved from the block. "IANA Ethernet Address block - Unicast Use"

## 14. References

### 14.1. Normative References

- [I-D.draft-ietf-mpls-in-udp]  
Xu, Sheth, Yong, Pignataro, Yongbing , and Li,  
"Encapsulating MPLS in UDP", May 2013.
- [I-D.draft-lasserre-nvo3-framework]  
Lasserre, M., Balus, F., Morin, T., Bitar, N., and Y.  
Rekhter, "Framework for DC Network Virtualization",  
September 2011.
- [I-D.draft-mahalingam-dutt-dcops-vxlan]  
Mahalingam, M., Dutt, D., Agarwal, P., Kreeger, L.,  
Sridhar, T., Bursell, M., and C. Wright, "VXLAN: A  
Framework for Overlaying Virtualized Layer 2 Networks  
over Layer 3 Networks", May 2013.
- [I-D.draft-singh-nvo3-nvgre-router-alert]  
Singh, K., Jain, P., Balus, F., and W. Henderickx, "NVGRE  
Router Alert Option", May 2013.
- [I-D.draft-singh-nvo3-vxlan-router-alert]  
Singh, K., Jain, P., Balus, F., and W. Henderickx, "VxLAN  
Router Alert Option", May 2013.
- [I-D.draft-sridharan-virtualization-nvgre]  
Sridharan, M., Duda, K., Greenberg, A., Lin, G., Pearson,  
M., Thaler, P., Tumuluri, C., Venkataramiah, N., and Y.  
Wang, "NVGRE: Network Virtualization using Generic Routing  
Encapsulation", February 2013.
- [RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y.,  
Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack  
Encoding", RFC 3032, January 2001.
- [RFC4023] Worster, T., Rekhter, Y., and E. Rosen, "Encapsulating  
MPLS in IP or Generic Routing Encapsulation (GRE)",  
RFC 4023, March 2005.
- [RFC4365] Rosen, E., "Applicability Statement for BGP/MPLS IP  
Virtual Private Networks (VPNs)", RFC 4365, February 2006.
- [RFC4379] Kompella, K. and G. Swallow, "Detecting Multi-Protocol  
Label Switched (MPLS) Data Plane Failures", RFC 4379,  
February 2006.

## 14.2. Informative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC4330] Mills, D., "Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI", RFC 4330, January 2006.

## Authors' Addresses

Pradeep Jain  
Nuage Networks  
755 Ravendale Drive  
Mountain View, CA 94043  
USA

Email: [pradeep@nuagenetworks.net](mailto:pradeep@nuagenetworks.net)

Kanwar Singh  
Nuage Networks  
755 Ravendale Drive  
Mountain View, CA 94043  
USA

Email: [kanwar@nuagenetworks.net](mailto:kanwar@nuagenetworks.net)

Florin Balus  
Nuage Networks  
755 Ravendale Drive  
Mountain View, CA 94043  
USA

Email: [florin@nuagenetworks.net](mailto:florin@nuagenetworks.net)

Wim Henderickx  
Alcatel-Lucent  
Copernicuslaan 50  
Antwerp 2018  
Belgium

Email: [wim.henderickx@alcatel-lucent.be](mailto:wim.henderickx@alcatel-lucent.be)

Vinay Bannai  
PayPal  
2211 N. First St,  
San Jose 95131  
USA

Email: [vbannai@paypal.com](mailto:vbannai@paypal.com)

Ravi Shekhar  
Juniper Networks  
1194 North Mathilda Ave.  
Sunnyvale, CA 94089  
USA

Email: [rshekhar@juniper.net](mailto:rshekhar@juniper.net)

Anil Lohiya  
Juniper Networks  
1194 North Mathilda Ave.  
Sunnyvale, CA 94089  
USA

Email: [alohiya@juniper.net](mailto:alohiya@juniper.net)





NVO3 Working Group  
Internet Draft  
Intended status: Standard Track

Tissa Senevirathne  
Norman Finn  
Samer Salam  
Deepak Kumar  
CISCO

Donald Eastlake  
Sam Aldrin  
Huawei

February 11, 2014

Expires: August 2014

NVO3 Fault Management  
draft-tissa-nvo3-oam-fm-00.txt

#### Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at  
<http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at  
<http://www.ietf.org/shadow.html>

This Internet-Draft will expire on August 11, 2014.

#### Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Abstract

This document specifies Fault Management solution for network virtualization overlay networks. Methods in this document follow the IEEE 802.1 CFM framework and reuse OAM tools where possible. Additional messages and TLVs are defined for IETF overlay OAM specific applications or where extensions beyond IEEE 802.1 CFM are required.

## Table of Contents

1. Introduction.....	3
2. Conventions used in this document.....	5
3. NVO3 OAM Layers.....	5
4. General Format of NVO3 OAM frames.....	6
4.1. NVO3 Shim.....	8
4.2. Identification of OAM frames.....	8
5. Maintenance Associations (MA) in NVO3.....	9
6. MEP Addressing.....	10
6.1. Use of MIP in NVO3.....	13
7. Continuity Check Message (CCM).....	15
8. NVO3 OAM Message Channel.....	17
8.1. NVO3 OAM Message header.....	17
8.2. IETF Overlay OAM Opcodes.....	18
8.3. Format of IETF Overlay OAM TLV.....	18
8.4. IETF Overlay OAM TLVs.....	19
8.4.1. Common TLVs between 8201Q CFM and IETF Overlay OAM..	19
8.4.2. IETF Overlay OAM Specific TLVs.....	20
8.4.3. OAM Application Identifier TLV.....	20
8.4.4. Out Of Band Reply Address TLV.....	22
8.4.5. Diagnostics Label TLV.....	22
8.4.6. Original Data Payload TLV.....	23
8.4.7. Flow Identifier (flow-id) TLV.....	24
8.4.8. Reflector Entropy TLV.....	24

9. Loopback Message.....	25
9.1. Loopback OAM Message format.....	25
9.2. Theory of Operation.....	26
9.2.1. Actions by Originator.....	26
9.2.2. Intermediate Devices.....	26
9.2.3. Destination Device.....	27
10. Path Trace Message.....	27
10.1. Theory of Operation.....	28
10.1.1. Action by Originator Device.....	28
10.1.2. Intermediate Device.....	29
10.1.3. Destination Device.....	30
11. Application of Continuity Check Message (CCM) in NVO3.....	30
11.1. CCM Error Notification.....	31
11.2. Theory of Operation.....	32
11.2.1. Actions by Originator Device.....	32
11.2.2. Intermediate Devices.....	32
11.2.3. Destination Device.....	33
12. Security Considerations.....	33
13. IANA Considerations.....	33
14. References.....	34
14.1. Normative References.....	34
14.2. Informative References.....	34
15. Acknowledgments.....	34
Appendix A. Base Mode for NVO3 OAM.....	35

## 1. Introduction

Conceptually, NVO3 architecture contains four separate layers, namely, Service (customer) Layer, Overlay Layer, Transport or Underlay Layer and Media Layer. Figure 1 below depicts the relationship between each of these layers.

Fault Monitoring, Fault Verification, Fault Isolation, Loss and delay measurements are integral part of NVO3 [nvo3oamReq]. These need to cover both unicast and multicast traffic streams.

For effective fault isolation, the overlay OAM solution should complement the OAM functions at adjacent layers, thereby leading to nested OAM model. Nested OAM allows operators to quickly and effectively troubleshoot and isolate data plane failures using a common OAM framework.

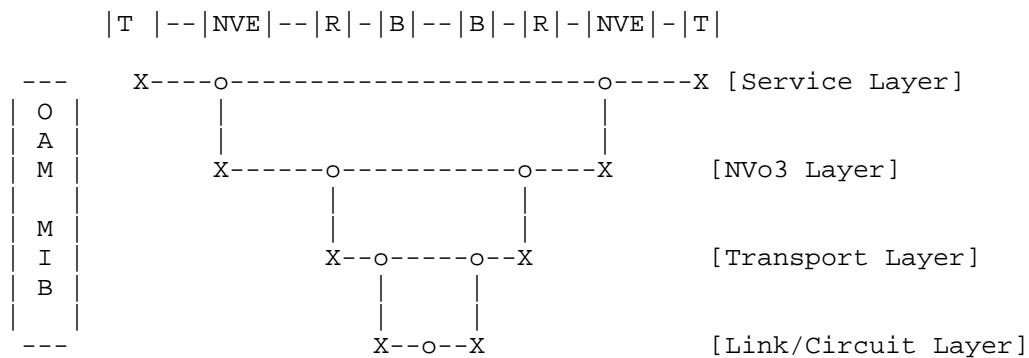
Common OAM message format and infrastructure makes it easier to accomplish nested OAM. IEEE Connectivity Fault Management (CFM) [8021Q] is widely used in the Ethernet world. The same technology has been extended by ITU-T [Y1731], Metro Ethernet Forum (MEF) and TRILL [TRILLFM].

A common OAM message channel can be shared between different technologies. This consistency between different OAM technologies promotes nested fault monitoring and isolation between technologies that share the same OAM framework.

This document uses the message format defined in IEEE 802.1ag Connectivity Fault Management (CFM) [8021Q] as the basis for the OAM messages.

This document presents the NVO3 OAM message structure, NVO3 frame identification and NV03 OAM tools. The NVO3 OAM Management Information Base (MIB) will be presented in a separate document.

The ITU-T Y.1731 [Y1731] standard utilizes the same messaging format as [8021Q] and for OAM messages where applicable. This document takes a similar stance and reuses [8021Q] and TRILL OAM [TRILLFM]. It is assumed that readers are familiar with [8021Q] and [Y1731].



X - Maintenance End Point (MEP)  
o - Maintenance Intermediate Point (MIP)

T - Tenant System      NVE - Network Virtualization Edge  
R - Router              B - Bridge

Figure 1 Layered OAM Architecture

## 2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RFC2119].

Acronyms used in the document include the following:

MP - Maintenance Point [8021Q]  
MEP - Maintenance End Point [8021Q]  
MIP - Maintenance Intermediate Point [8021Q]  
MA - Maintenance Association [8021Q]  
MD - Maintenance Domain [8021Q]  
CCM - Continuity Check Message [8021Q]  
LBM - Loop Back Message [8021Q]  
PTM - Path Trace Message  
MTV - Multi-destination Tree Verification Message  
ECMP - Equal Cost Multipath  
ISS - Internal Sub Layer Service [8021Q]  
VNI - Virtual Network Instance [NVO3FRM]  
NVE - Network Virtual Edge [NVO3FRM]  
SAP - Service Access Point [8021Q]

## 3. NVO3 OAM Layers

Figure 1 above depicts different layers within NVO3. Each of these layers has a unique scope within the common framework. In this section, we define functionality of each of these layers

**Service Layer:** Service Layer carries customer or user traffic. It is originated at Tenant systems and terminates at other Tenant system(s) within the same NVO3 context (VNI).

NVO3 Layer: NVO3 Layer carries service Layer traffic encapsulated in NVO3 format. NVO3 Layer originates at an NVE and terminates at another NVE.

Transport Layer: Transport Layer carries NVO3 Layer traffic encapsulated in its data format. The transport Layer can be IP, MPLS or any other protocol.

Link Layer: This is also known as circuit layer. It carries Transport Layer traffic in a specific manner from one device to another. Ethernet is an example of link layer.

#### 4. General Format of NVO3 OAM frames

For accurate monitoring and/or diagnostics, OAM Messages are required to follow the same data path as corresponding user packets. Additionally, NVEs are required to identify NVO3 frames and act on them, in addition to preventing the overlay OAM packets from leaking outside of the NVO3 domain [NVO3OAMREQ]. This document defines the format of the NVO3 overlay OAM messages.

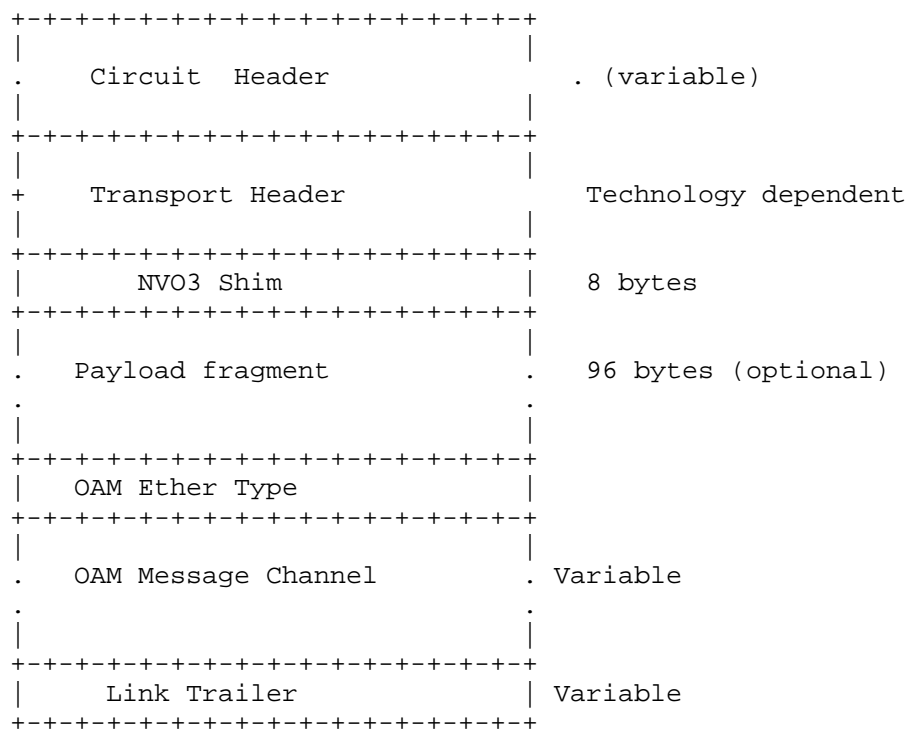


Figure 2 Format of NV03 Overlay OAM Messages

Link Header: Media-dependent header. For Ethernet, this includes Destination MAC, Source MAC, VLAN (optional) and EtherType fields.

Transport Header: Header of the Transport Layer e.g. IP , MPLS, TRILL etc.

NVO3 Shim: This is a fixed sized field (size TBD 8 bytes [vxLAN]) that carries NVO3 specific information. Fields in this header will be utilized to identify OAM frames and other NVO3 specific operations. Please see section 4.2 below of NVO3 OAM specific operations.

**Payload Fragment:** This is an optional field. When included it has a fixed size of 96 bytes. The least significant bits of the field **MUST** be padded with zeros, up to 96 bytes, when the payload fragment is



less than 96 bytes. The Payload Fragment field starts with the Inner.MacDA.

**OAM Ether Type:** OAM Ether Type is 16-bit EtherType that identifies the OAM Message channel that follows. This document specifies using the EtherType 0x8902 allocated for [8021Q] for this purpose. Identifying the OAM Message Channel with a dedicated EtherType allows the easy identification of the beginning of the OAM message channel across multiple standards.

**OAM Message Channel:** This is a variable size section that carries OAM related information. The message format defined in [8021Q] will be reused.

**Link Trailer:** Media-dependent trailer. For Ethernet, this is the FCS (Frame Check Sequence).

#### 4.1. NVO3 Shim

NVO3 Shim is an 8 octet vector that carries series of flags and additional information [vxLAN]. Each of the flags identifies a specific operation.

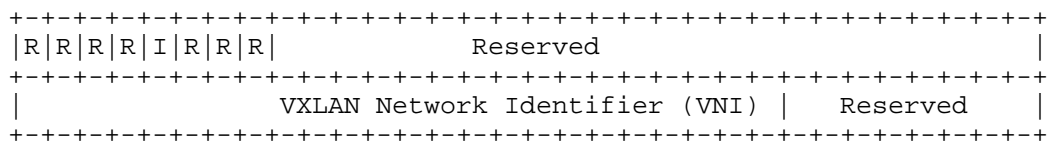


Figure 3 NVO3 Shim

#### 4.2. Identification of OAM frames

Implementations that comply with this document MUST utilize "O" flag to identify NVO3 OAM frames. The "O" flag MUST NOT BE utilized for forwarding decisions such as the selection of which ECMP path to use.

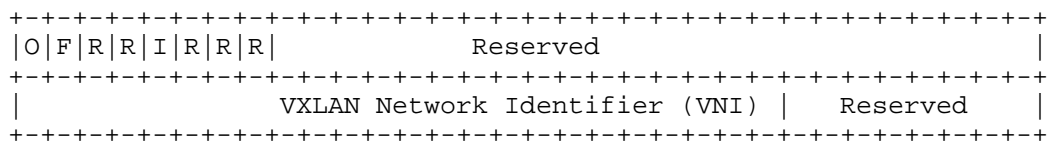


Figure 4 NVO3 shim with the "O" Flag

O (1 bit) - Indicates this is a possible OAM frame and is subject to specific handling as specified in this document.

F (1 bit) - When set to 1 indicates that 96 octets of payload fragments is included immediately after the NVO3 shim. When set 0 indicates that OAM Ethertype 0x8902 immediately follows the NVO3 shim.

All other fields carry the same meaning as defined in [vXLAN].

## 5. Maintenance Associations (MA) in NVO3

[8021Q] defines a maintenance association as a logical relationship between a group of nodes. Each Maintenance Association (MA) is identified with a unique MAID of 48 bytes [8021Q]. CCM and other related OAM functions operate within the scope of an MA. The definition of MA is technology independent. MA is encoded in the technology independent part of the OAM message Hence the MAID, as defined in [8021Q], can be utilized for NVO3 OAM, without modifications. This also allows us to utilize CCM and LBM messages defined in [8021Q], as is.

In NVO3, an MA may contain two or more NVEs (MEPs). For unicast, it is likely that the MA contains exactly two MEPs that are the two endpoints of the flow. For multicast, the MA may contain two or more MEPs.

For NVO3, in addition to all of the standard CFM MIB [8021Q] definitions, each MEP's MIB contains one or more flow definitions corresponding to the set of flows that the MEP monitors. Flow entropy specifies the VNI within the NVE.

MEPs can be created per VNI within the NVE.

We propose to augment the [8021Q] MIB to add NVO3 specific information. Figure 5, below depicts the augmentation of the CFM MIB to add the NVO3 specific Flow Entropy.

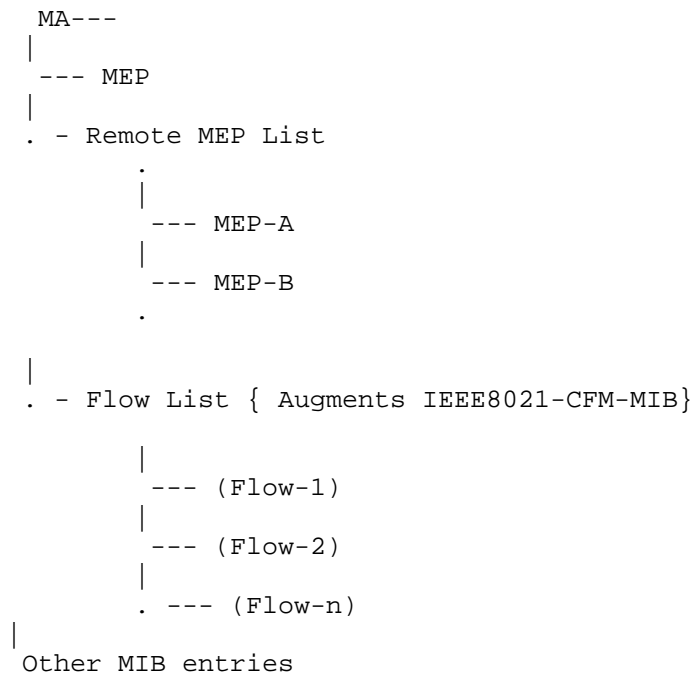


Figure 5 Correlation of NVO3 augmented MIB

The flow list contains multiple flows. Each flow contains a VNI and optional data payload. There can be more than one flow for a given VNI with different data payloads e.g. unicast vs. multicast or unicast with different data payloads.

## 6. MEP Addressing

In IEEE 802.1ag [8021Q], OAM messages address the target MEP by utilizing a unique MAC address. In NVO3, MEPs are created per VNI, MEPs are addressed by combination of Transport Layer address of the NVE and the VNI.

At the MEP, OAM packets go through a hierarchy of op-code de-multiplexers. The op-code de-multiplexers channel the incoming OAM packets to the appropriate message processor (e.g. LBM) The reader may refer to Figure 6 below for a visual depiction of these different de-multiplexers.

1. Identify the packets that need OAM processing at the Local Device as specified in Section 4.2.
  - a. Identify the MEP that is associated with the VNI.
2. The MEP then validates the MD-LEVEL
  - a. Redirect to MD-LEVEL De-multiplexer
3. MD-LEVEL de-multiplexer compares the MD-Level of the packet against the MD level of the local MEPs. (Note: there can be more than one MEP at the same MD-Level but belonging to different MAs)
  - a. If the packet MD-LEVEL is equal to the configured MD-LEVEL of the MEP, then pass to the Opcode de-multiplexer
  - b. If the packet MD-LEVEL is less than the configured MD-LEVEL of the MEP, discard the packet
  - c. If the packet MD-LEVEL is greater than the configured MD-LEVEL of the MEP, then pass on to the next higher MD-LEVEL de-multiplexer, if available. Otherwise, if no such higher MD-LEVEL de-multiplexer exists, then forward the packet as normal data.
4. Opcode De-multiplexer compares the opcode in the packet with the supported opcodes
  - a. If Op-code is CCM, LBM, LBR, PTM, PTR, MTVM, MTVR, then pass on to the correct Processor
  - b. If Op-code is Unknown, then discard.

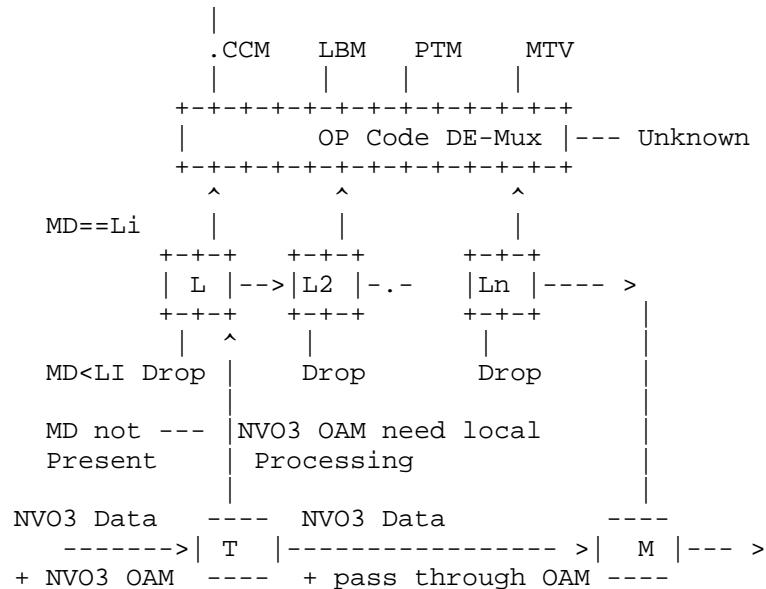


Figure 6 OAM De-Multiplexers at MEP for active SAP

T : Denotes Tap, that identifies OAM frames that need local processing. These are the packets with OAM flag set AND OAM Ether type is present after the flow entropy of the packet

M : Is the post processing merge, merges data and OAM messages that are passed through. Additionally, the Merge component ensures, as explained earlier, that OAM packets are not forwarded out as native frames.

L : Denotes MD-Level processing. Packets with MD-Level less than the Level will be dropped. Packets with equal MD-Level are passed on to the opcode de-multiplexer. Others are passed on to the next level MD processors or eventually to the merge point (M).

NOTE: LBM, MTV and PT are not subject to MA de-multiplexers. These packets do not have an MA encoded in the packet. Adequate response can be generated to these packets, without loss of functionality, by any of the MEPS.

### 6.1. Use of MIP in NVO3

Maintenance Intermediate Points (MIP) are mainly used for fault isolation. Link Trace Messages in [8021Q] utilize a well-known multicast MAC address and MIPs generate responses to Link Trace messages. Response to Link Trace messages or lack thereof can be used for fault isolation in NVO3.

As explained in section 10. below, a TTL expiry approach will be utilized for fault isolation and path tracing. The approach is very similar to the well-known IP trace-route approach. Hence, explicit addressing of MIPs is not required for the purpose of fault isolation.

Any given NVO3 device can have multiple MIPs located within the device. As such, a mechanism is required to identify which MIP should respond to an incoming OAM message.

A similar approach to that presented above for MEPs can be used for MIP processing. It is important to note that "M", the merge block of a MIP, does not prevent OAM packets leaking out as native frames. On edge interfaces, MEPs MUST be configured to prevent the leaking of overlay OAM packets out of the NVO3 domain.

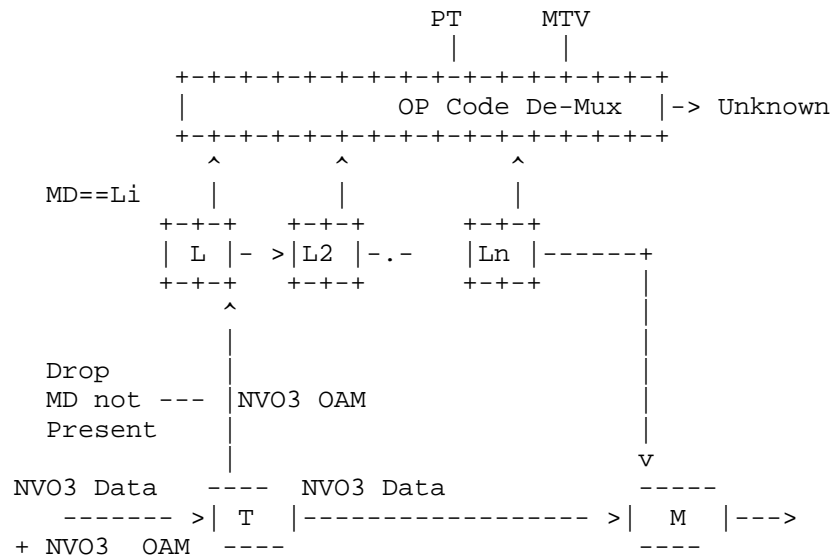


Figure 7 OAM De-Multiplexers at MIP for active SAP

T: TAP processing for MIP. All packets with OAM flag set are captured.

L : MD Level Processing, Packet with matching MD Level are "copied" to the Opcode de-multiplexer and original packet is passed on to the next MD level processor. Other packets are simply passed on to the next MD level processor, without copying to the OP code de-multiplexer.

M : Merge processor, merge OAM packets to be forwarded along with the data flow.

Packets that carry Path Trace Message (PTM) or Multi-destination Tree Verification (MTV) OpCodes are passed on to the respective processors.

Packets with unknown OpCodes are counted and discarded.

## 7. Continuity Check Message (CCM)

CCMs are used to monitor connectivity and configuration errors. [8021Q] monitors connectivity by having a MEP listening to periodic CCM messages received from its remote MEP partners in the MA. An [8021Q] MEP identifies cross-connect errors by comparing the MAID in the received CCM message with the MEP's local MAID. The MAID [8021Q] is a 48-byte field that is technology independent. Similarly, the MEPID is a 2-byte field that is independent of the technology. Given this generic definition of CCM fields, CCM as defined in [8021Q] can be utilized in NVO3 with no changes. NVO3 specific information may be carried in CCMs when encoded using IETF overlay specific TLVs or sub-TLVs. This is possible since CCMs are capable of carrying optional TLVs.

Unlike classical Ethernet environments with spanning tree, NVO3 supports multipath forwarding. The path taken by a packet depends on the Transport header and other parts of the packet. The Maintenance Association identifies the interested end-points (MEPs) of a given monitored path. For unicast there are only two MEPs per MA. For multicast there can be two or more MEPs in the MA. The Flow (i.e VNI and other parameters) values of the monitored flows are defined within the MA. CCM transmit logic will utilize these flow entropy values when constructing the CCM packets. Please see below for the theory of operation of CCM.

The CFM MIB of [8021Q] will be augmented with the definition of flow-entropy. Please see [TBDMIBD] for these and other NVO3 related OAM MIB definitions. The figure below depicts the correlation between MA, CCM and the flow.



```

      MA---
      |
      --- MEP
      |
      . - Remote MEP List
          .
          |
          --- MEP-A
          |
          --- MEP-B
          .

      |
      . - Flow List {Augments IEEE8021-CFM-MIB}
          |
          --- (Flow-1) {note we have to define
                        destination NVE with
          --- (Flow-2)  the flow entropy discuss}
          |
          . --- (Flow-n)

      |
      . - CCM
          |
          --- (standard 8021ag entries)
          |
          --- (TTL) { Augments IEEE8021-CFM-MIB}
          |
          --- (Other TBD NVO3 OAM specific entries)
                        {Augmented}

      |
      .
      |
      - Other MIB entries

```

Figure 8 Augmentation of CCM MIB in NVO3

NOTE: Flow entropy field contain VNI and flow specific information that affect the path selection and forwarding.

In a multi-pathing environment, a Flow - by definition - is unidirectional. A question may arise as to what flow entropy should be used in the response. CCMs are unidirectional and have no explicit

reply; as such, the issue of the response flow entropy does not arise. In the transmitted CCM, each MEP reports local status using the Remote Defect Indication (RDI) flag. Additionally, a MEP may raise SNMP TRAPS [TBDMIB] as Alarms when a connectivity failure occurs.

## 8. NVO3 OAM Message Channel

The NVO3 OAM Message Channel can be divided into two parts: NVO3 OAM Message header and NVO3 OAM Message TLVs. Every OAM Message MUST contain a single OAM message header and a set of one or more specified OAM Message TLVs.

### 8.1. NVO3 OAM Message header

As discussed earlier, a common messaging framework between [8021Q], NVO3, and other similar standards such as Y.1731 can be accomplished by re-using the OAM message header defined in [8021Q].

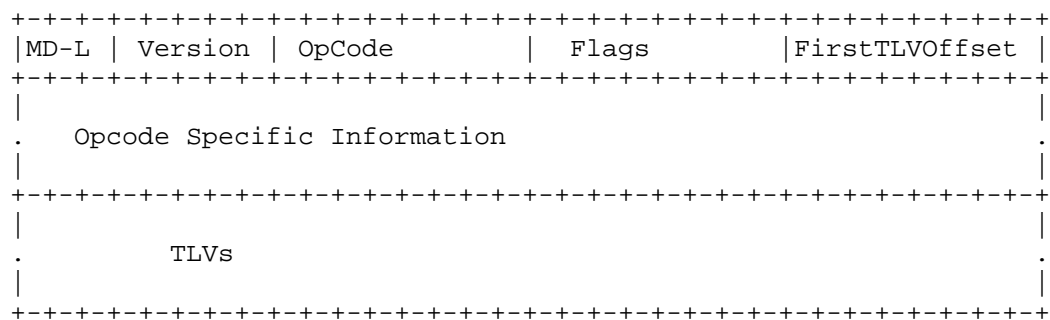


Figure 9 OAM Message Format

- o MD-L: Maintenance Domain Level (3 bits). Identifies the maintenance domain level. For NVO3, in general, this field is set to a single value. If using Base Mode operations as defined in Appendix B, this field is set to 3. However, future extensions of NVO3, for example to support hierarchy, may create different MD-LEVELs and MD-L field must be appropriately set in those scenarios. (Please refer to [8021Q] for the definition of MD-Level)

- o Version: Indicates the version (5 bits), as specified in [8021Q]. This document does not require changing the Version defined in [8021Q].
- o Flags: Includes operational flags (1 byte). The definition of flags is Opcode-specific and is covered in the applicable sections.
- o FirstTLVOffset: Defines the location of the first TLV, in bytes, starting from the end of the FirstTLVOffset field (1 byte). (Refer to [8021Q] for the definition of the FirstTLVOffset.)

MD-L, Version, Opcode, Flags and FirstTLVOffset fields collectively are referred to as the OAM Message Header.

The Opcode specific information section of the OAM Message may contain Session Identification number, time-stamp, etc.

## 8.2. IETF Overlay OAM Opcodes

The following Opcodes are defined for IETF Overlay OAM. Each of the Opcodes indicates a separate type of OAM message. Details of the messages are presented in the related sections.

IETF OAM Message Opcodes:

TBD-64 : Path Trace Reply  
TBD-65 : Path Trace Message  
TBD-66 : Multicast Tree Verification Reply  
TBD-67 : Multicast Tree Verification Message

## 8.3. Format of IETF Overlay OAM TLV

The same TLV format as defined in section 21.5.1 of [8021Q] is used for IETF Overlay OAM. The following figure depicts the general format of a IETF Overlay OAM TLV:

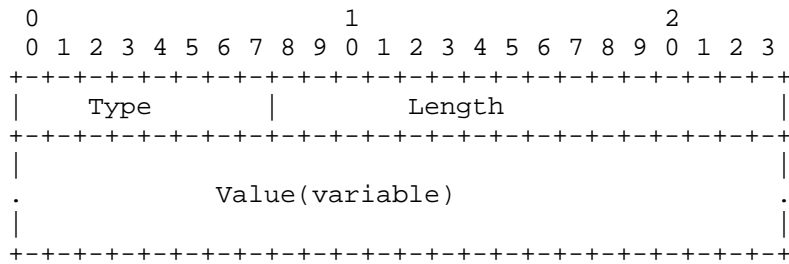


Figure 10 NVO3 OAM TLV

Type (1 octet): Specifies the Type of the TLV (see sections 8.4. for TLV types).

Length (2 octets): Specifies the length of the 'Value' field in octets. Length of the 'Value' field can be either zero or more octets.

Value (variable): The length and the content of this field depend on the type of the TLV. Please refer to applicable TLV definitions for the details.

Semantics and usage of Type values allocated for TRILL OAM purpose are defined by this document and other future related documents.

#### 8.4. IETF Overlay OAM TLVs

Overlay related TLVs are defined in this section. Previously defined [8021Q] TLVs are reused, where applicable. Block of 32 TLVs are allocated for the purpose of IETF defined standards

##### 8.4.1. Common TLVs between 8201Q CFM and IETF Overlay OAM

The following TLVs are defined in [8021Q]. We propose to re-use them where applicable. The format and semantics of the TLVs are as defined in [8021Q].

Type	Name of TLV in [8021Q]
0	End TLV
1	Sender ID TLV
2	Port Status TLV
3	Data TLV

- 4     Interface Status TLV
- 5     Reply Ingress TLV
- 6     Reply Egress TLV
- 7     LTM Egress Identifier TLV
- 8     LTR Egress Identifier TLV
- 9-30 Reserved
- 31    Organization Specific TLV

#### 8.4.2. IETF Overlay OAM Specific TLVs

As indicated above, a block of 32 TLVs will be requested to be reserved for IETF OAM purposes. Listed below is a summary of the IETF Overlay OAM TLVs and their corresponding codes. Format and semantics of OAM TLVs are defined in subsequent sections.

Type	TLV Name
-----	-----
TBD-TLV-64	OAM Application Identifier
TBD-TLV-65	Out of Band IP Address
TBD-TLV-66	Original Payload
TBD-TLV-67	Diagnostic VLAN
TBD-TLV-68	scope
TBD-TLV-69	Previous Device address
TBD-TLV-70	Next Hop Device List (ECMP)
TBD-TLV-71	Multicast Receiver Availability
TBD-TLV-72	Flow Identifier
TBD-TLV-73	Reflector Entropy
TBD-TLV-74 to TBD-TLV-95	Reserved

#### 8.4.3. OAM Application Identifier TLV

OAM Application Identifier TLV carries Overlay OAM application specific information. The Overlay OAM Application Identifier TLV MUST always be present and MUST be the first TLV in the OAM messages. Messages that do not include the OAM Application Identifier TLV as the first TLV MUST be discarded.

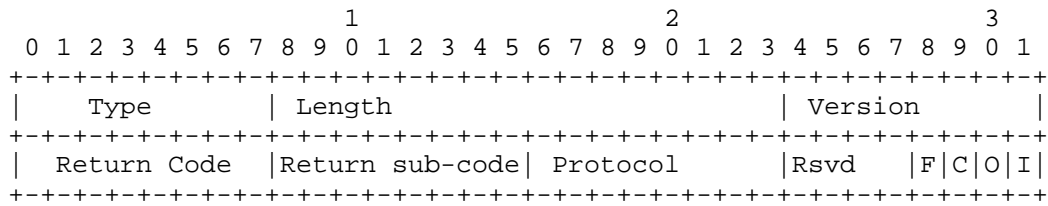


Figure 11 OAM Application Identifier TLV

Type (1 octet) = TBD-TLV-64 indicate that this is the IETF Overlay OAM Application Identifier TLV

Length (2 octets) = 6

IETF Overlay OAM Version (1 Octet), currently set to zero. Indicates the Overlay OAM version. IETF Overlay OAM version can be different than the [8021Q] version.

Return Code (1 Octet): Set to zero on requests. Set to an appropriate value in response messages.

Return sub-code (1 Octet): Return sub-code is set to zero on transmission of request message. Return sub-code identifies categories within a specific Return code. Return sub-code MUST be interpreted within a Return code.

Protocol: This indicates the overlay protocol on which the OAM is applied. In this document we cover NVO3

0 : TRILL

1 : NVO3

2 - 255 : reserved

F (1 bit): Final flag, when set, indicates this is the last response.

C (1 bit): Label error, if set indicates that the label (VLAN) in the flow entropy is different than the label included in the diagnostic TLV. This field is ignored in request messages and MUST only be interpreted in response messages.

O (1 bit): If set, indicates, OAM out-of-band response requested.

I (1 bit): If set, indicates, OAM in-band response requested.

NOTE: When both O and I bits are set to zero, indicates that no response is required (silent mode). User MAY specify both O and I or one of them or none. When both O and I bits are set response is sent both in-band and out-of-band.

#### 8.4.4. Out Of Band Reply Address TLV

Out of Band Reply Address TLV specifies the address to which an out of band OAM reply message MUST be sent. When O bit in the Application Identifier TLV is not set, Out of Band Reply Address TLV is ignored.

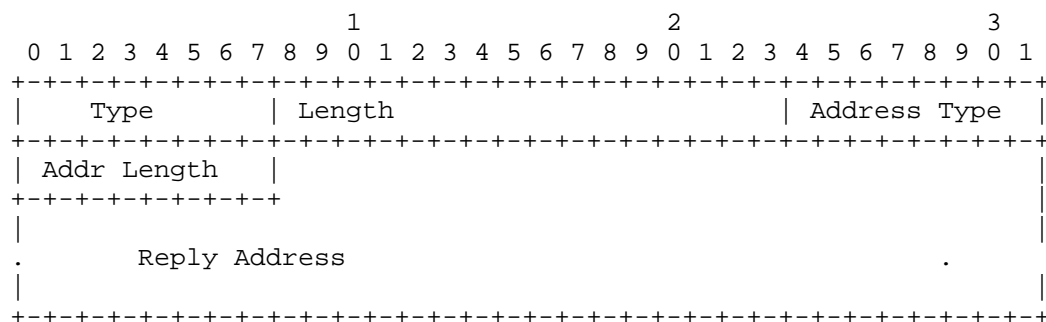


Figure 12 Out of Band IP Address TLV

Type (1 octet) = TBD-TLV-65

Length (2 octets) = Variable. Minimum length is 2.

Address Type (1 Octet): 0 - IPv4. 1 - IPv6. All other values reserved.

Addr Length (1 Octet). 4 - IPv4. 16 - IPv6,

Reply Address (variable): Address where the reply needed to be sent. Length depends on the address specification.

#### 8.4.5. Diagnostics Label TLV

Diagnostic label specifies the VNI which the OAM messages are generated. Receiving device MUST compare the VNI of the NVO3 shim to that of Diagnostic TLV. Label Error Flag in the response MUST be set when the two VLANs do not match.

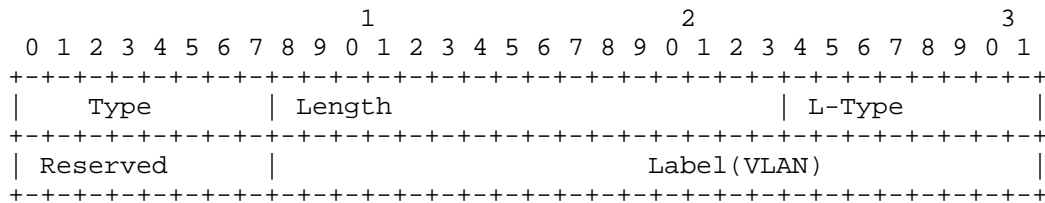


Figure 13 Diagnostic TLV

Type (1 octet) = TBD-TLV-66 indicates that this is the Diagnostic TLV

Length (2 octets) = 5

L-Type (Label type, 1 octet)

0- indicate 802.1Q 12 bit VLAN.

2 - 24 bit ISID

3 - VNI 24 bits

Label (24 bits): Either 12 bit VLAN or 24 bit ISID or 24 bit VNI.

Devices should perform Label error checking when Label TLV is not included in the OAM message.

#### 8.4.6. Original Data Payload TLV

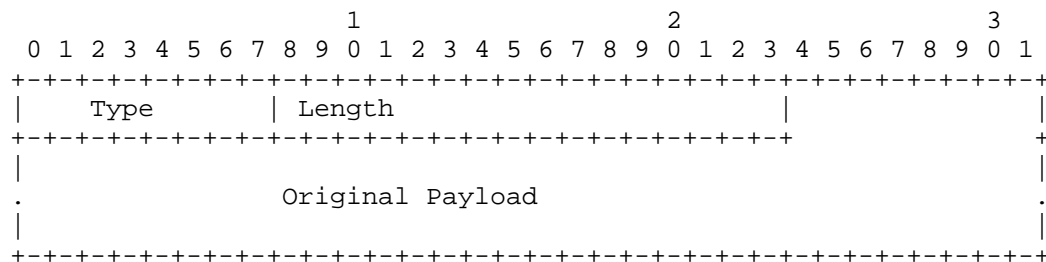


Figure 14 Original Data Payload TLV

Type (1 Octet) = TBD-TLV-67



Length (2 octets) = variable

#### 8.4.7. Flow Identifier (flow-id) TLV

Flow Identifier (flow-id) uniquely identifies a specific flow. The flow-id value is unique per MEP and needs to be interpreted as such.

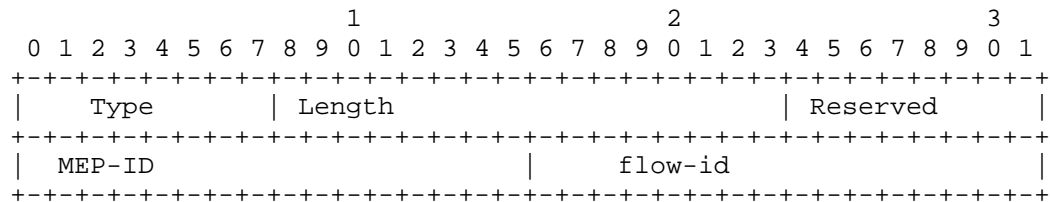


Figure 15 Flow Identifier TLV

Type (1 octet) = TBD-TLV-72

Length (2 octets) = 5.

Reserved (1 octet) set to 0 on transmission and ignored on reception.

MEP-ID (2 octets) = MEP-ID of the originator [8021Q].

Flow-id (2 octets) = uniquely identifies the flow per MEP. Different MEPs may allocate the same flow-id value. The {MEP-ID, flow-id} pair is globally unique.

Inclusion of the MEP-ID in the flow-id TLV allows inclusion of MEP-ID for messages that do not contain MEP-ID in OAM header. Applications may use MEP-ID information for different types of troubleshooting.

#### 8.4.8. Reflector Entropy TLV

Reflector Entropy TLV is an optional TLV. This TLV, when present, tells the responder to utilize the Reflector Entropy specified within the TLV as the flow-entropy of the response message.

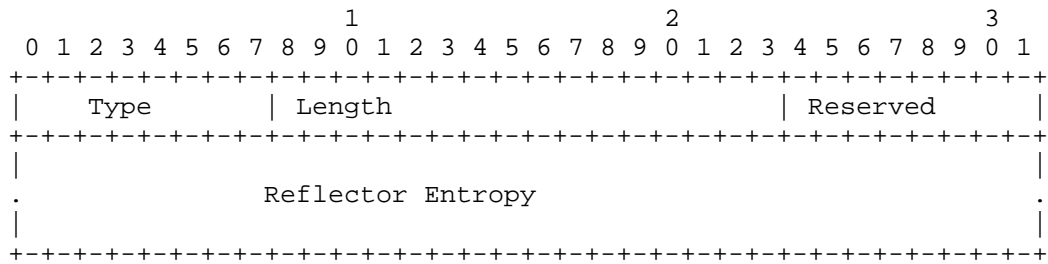


Figure 16 Reflector Entropy TLV

Type (1 octet) = TBD-TLV-73 Reflector Entropy TLV.

Length (1 octet) = 97.

Reserved (1 octet) = set to zero on transmission and ignored by the recipient.

Reflector Entropy (96-octet) = Flow Entropy to be used by the responder. May be padded with zero if the desired flow entropy is less than 96 octets.

## 9. Loopback Message

### 9.1. Loopback OAM Message format

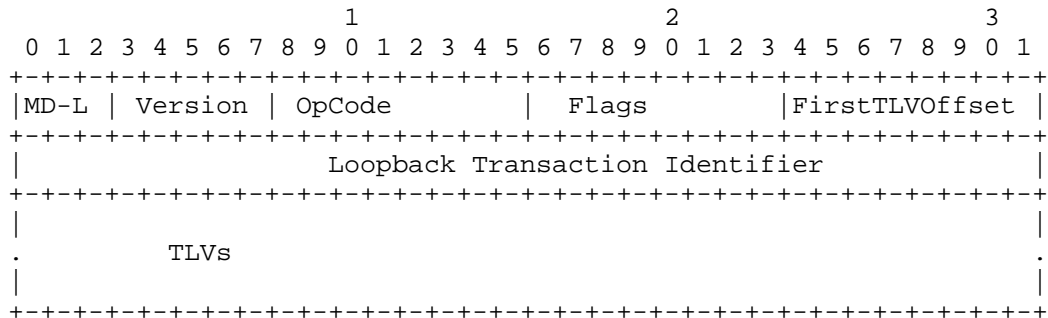


Figure 17 Loopback OAM Message Format

The above figure depicts the format of the Loopback Request and response messages as defined in [8021Q]. The Opcode for Loopback Message is set to 65 and the Opcode for the Reply Message is set to 64. The Session Identification Number is a 32-bit integer that allows the requesting Device to uniquely identify the corresponding session. Responding Device, without modification, MUST echo the received "Loopback Transaction Identifier" number..

## 9.2. Theory of Operation

### 9.2.1. Actions by Originator

Identifies the destination NVE address based on user specification or based on the specified destination, VNI and MAC

Constructs the flow entropy based on user specified parameters or implementation specific default parameters.

Constructs the OAM header: sets the opcode to Loopback message type (3). Assign applicable Loopback Transaction Identifier number for the request.

The Overlay OAM Version TLV MUST be included and with the flags set to applicable values.

Include following OAM TLVs, where applicable

- o Out-of-band Reply address TLV
- o Diagnostic Label TLV
- o Sender ID TLV

Specify the Transport Layer parameters based on user inputs or default settings.

Dispatch the OAM frame for transmission.

Devices may continue to retransmit the request at periodic intervals, until a response is received or the re-transmission count expires. At each transmission Session Identification number MUST be incremented.

### 9.2.2. Intermediate Devices

Intermediate Devices forward the frame as a normal data frame and no special handling is required.

### 9.2.3. Destination Device

If the Loopback message is addressed to the local Device and satisfies the OAM identification criteria specified in section 4.2. then, the Device data plane forwards the message to the CPU for further processing.

The OAM application layer further validates the received OAM frame by checking for the presence of OAM-Ethertype and the MD Level. Frames that do not contain OAM-Ethertype MUST be discarded.

Construction of the OAM response:

OAM application encodes the received Transport header, NVO3 shim and payload fragment (when present) in the Original payload TLV and includes it in the OAM message.

Set the Return Code and Return sub code to applicable values. Update the OAM opcode to 2 (Loopback Message Reply)

Optionally, if the VNI identifier value of the received differs from the value specified in the diagnostic Label, set the Label Error Flag on OAM Application Identifier TLV.

Include the sender ID TLV (1)

If in-band response was requested, dispatch the frame to the NVO3 data plane with request-originator Transport Layer address as the destination address.

If out-of-band response was requested, dispatch the frame to the IP forwarding process.

## 10. Path Trace Message

The primary use of the Path Trace Message is for fault isolation. It may also be used for plotting the path taken from a given Device to another Device.

[8021Q] accomplishes the objectives of the NVO3 Path Trace Message using Link Trace Messages. Link Trace Messages utilize a well-known multicast MAC address. However, in NVO3 the transport Layer can be different technologies such as IP or MPLS etc. Hence, a definition of a new Path Trace message format is required for Overlay OAM. The Path Trace message is defined for the purpose.

The Path Trace Message has the same format as Loopback Message, but utilizes a different opcode set. Opcode for Path Trace Reply Message is TBD-64 and Request Message is TBD-65

Operation of the Path Trace message is identical to the Loopback message except that it is first transmitted with a TTL field value of 1. The sending device expects a Time Expiry Return-Code from the next hop or a successful response. If a Time Expiry Return-code is received as the response, the originator Device records the information received from intermediate node that generated the Time Expiry message and resends the message by incrementing the previous TTL value by 1. This process is continued until, a response is received from the destination Device or Path Trace process timeout occur or TTL reaches a configured maximum value.

#### 10.1. Theory of Operation

##### 10.1.1. Action by Originator Device

Identifies the destination NVE address based on user specification or based on the specified destination, VNI and MAC

Constructs the NVO3 shim and the flow based on user specified parameters or implementation specific default parameters.

Construct the OAM header: Set the opcode to Path Trace Request message type (TBD-65). Assign an applicable Session Identification number for the request. Return-code and sub-code MUST be set to zero.

The OAM Application Identifier TLV MUST be included and set the flags to applicable values.

Include following OAM TLVs, where applicable

- o Out-of-band IP address TLV
- o Diagnostic Label TLV
- o Include the Sender ID TLV

Specify the TTL of the transport header as 1 for the first request.

Dispatch the OAM frame to the data plane for transmission.

A Device (MEP) may continue to retransmit the request at periodic intervals, until a response is received or the re-transmission count expires. At each new re-transmission, the Session Identification

number MUST be incremented. Additionally, for responses received from intermediate devices (MIP), the device address and interface information MUST be recorded.

#### 10.1.2. Intermediate Device

Path Trace Messages transit through Intermediate devices transparently, unless Hop-count has expired.

OAM application layer further validates the received OAM frame by examining the presence of NVO3 OAM Flag and OAM-Ethertype and by examining the MD Level. Frames that do not contain OAM-Ethertype MUST be discarded.

Construction of the OAM response:

OAM application encodes the received Transport header and flow entropy in the Original payload TLV and include it in the OAM message.

Set the Return Code to (2) "Time Expired" and Return sub code to zero (0). Update the OAM opcode to 64 (Path Trace Message Reply).

If the VNI identifier value of the received OAM message differs from the value specified in the diagnostic Label, set the Label Error Flag on OAM Application Identifier TLV.

Include following TLVs

Reply Ingress TLV (5)

Reply Egress TLV (6)

Interface Status TLV (4)

Sender ID TLV (1)

If Label error detected, set C flag (Label error detected) in the version.

If in-band response was requested, dispatch the frame to the NVO3 data plane with request-originator Transport address as the destination address.

If out-of-band response was requested, dispatch the frame to the standard IP forwarding process.

#### 10.1.3. Destination Device

Processing is identical to Loop back response processing in section 9.2.3. with the exception that OAM Opcode is set to Path Trace Reply (64).

### 11. Application of Continuity Check Message (CCM) in NVO3

Section 7. provides an overview of CCM Messages defined in [8021Q] and how they can be used within the NVO3 OAM. This section, presents the application and Theory of Operations of CCM within the NVO3 OAM framework. Readers are referred to [8021Q] for CCM message format and applicable TLV definitions and usages. Only the NVO3 specific aspects are explained below.

In NVO3, between any two given MEPs there can be multiple potential paths. Whereas in [8021Q], there is always a single path between any two MEPs at any given time. It is important that solutions to have the ability to monitor continuity over one or more paths.

CCM Messages are uni-directional, such that there is no explicit response to a received CCM message. Connectivity fault status is indicated by setting the applicable flags (e.g. RDI) of the CCM messages transmitted by an MEP.

It is important that the solution presented in this document accomplishes the requirements specified in [NVO3OAMREQ] within the framework of [8021Q] in a straightforward manner and with minimum changes. Section 8 above defines multiple flows within the CCM object, each corresponding to a flow that a given MEP wishes to monitor.

Receiving MEPs do not cross check whether a received CCM belongs to a specific flow from the originating MEP. Any attempt to track status of individual flows may explode the amount of state information that any given MEP has to maintain.

The obvious question arises: How does the originating device know which flow or flows are at fault?

This is accomplished with a combination of the RDI flag in the CCM header, flow-id TLV, and SNMP Notifications (Traps). Section 11.1 below discusses the procedure.

### 11.1.1. CCM Error Notification

Each MEP transmits 4 CCM messages per each flow. ([8021Q] detects CCM fault when 3 consecutive CCM messages are lost). Each CCM Message has a unique sequence number and unique flow-identifier. The flow identifier is included in the OAM message via flow-id TLV.

When an MEP notices a CCM timeout from a remote MEP ( MEP-A), it sets the RDI flag on the next CCM message it generates. Additionally, it logs and sends SNMP notification that contain the remote MEP Identification, flow-id and the Sequence Number of the last CCM message it received and if available, the flow-id and the Sequence Number of the first CCM message it received after the failure. Each MEP maintains a unique flow-id per each flow, hence the operator can easily identify flows that correspond to the specific flow-id.

The following example illustrates the above.

Assume there are two MEPs, MEP-A and MEP-B.

Assume there are 3 flows between MEP-A and MEP-B.

Let's assume MEP-A allocates sequence numbers as follows

Flow-1 Sequence={1,2,3,4,13,14,15,16,... } flow-id=(1)

Flow-2 Sequence={5,6,7,8,17,18,19,20,... } flow-id=(2)

Flow-3 Sequence={9,10,12,11,21,22,23,24,... } flow-id=(3)

Let's Assume Flow-2 is at fault.

MEP-B, receives CCM from MEP-A with sequence numbers 1,2,3,4, but did not receive 5,6,7,8. CCM timeout is set to 3 CCM intervals in [8021Q]. Hence MEP-B detects the error at the 8'th CCM message. At this time the sequence number of the last good CCM message MEP-B has received from MEP-A is 4 and flow-id of the last good CCM Message is (1). Hence MEP-B will generate a CCM error SNMP notification with MEP-A and Last good flow-id (1) and sequence number 4.

When MEP-A switches to flow-3 after transmitting flow-2, MEP-B will start receiving CCM messages. In the foregoing example it will be CCM message with Sequence Numbers 9,10,11,12,21 and so on. When in receipt of a new CCM message from a specific MEP, after a CCM timeout, the NVO3 OAM will generate an SNMP Notification of CCM resume with remote MEP-ID and the first valid flow-id and the



Sequence number after the CCM timeout. In the foregoing example, it is MEP-A, flow-id (3) and Sequence Number 9.

The remote MEP list under the CCM MIB Object is augmented to contain "Last Sequence Number", flow-id and "CCM Timeout" variables. Last Sequence Number and flow-id are updated every time a CCM is received from a remote MEP. CCM Timeout variable is set when the CCM timeout occurs and is cleared when a CCM is received.

## 11.2. Theory of Operation

### 11.2.1. Actions by Originator Device

Derive the VNI and entropy based on flow entropy specified in the CCM Management object.

Construct the NVO3 CCM OAM header as specified in [8021Q].

OAM Version TLV MUST be included as the first TLV and set the flags to applicable values.

Include other TLVs specified in [8021Q]

Include the following optional OAM TLVs, where applicable

- o Sender ID TLV

Specify the TTL of the NVO3 data frame per user specification or utilize an applicable Hop count value.

Dispatch the OAM frame to the NVO3 data plane for transmission.

An MEP transmits a total of 4 requests, each at CCM retransmission interval. At each transmission the Session Identification number MUST be incremented by one.

At the 5'th retransmission interval, flow entropy of the CCM packet is updated to the next flow entropy specified in the CCM Management Object. If current flow entropy is the last flow entropy specified, move to the first flow entropy specified and continue the process.

### 11.2.2. Intermediate Devices

Intermediate devices forward the frame as a normal data frame and no special handling is required.

### 11.2.3. Destination Device

If the CCM Message is addressed to the local MEP or multicast and satisfies OAM identification methods specified in sections 4.2. then the data plane forwards the message to the CPU for further processing.

The OAM application layer further validates the received OAM frame by examining the presence of OAM-Ethertype at the end of the flow entropy. Frames that do not contain OAM-Ethertype at the end of the flow entropy MUST be discarded.

Validate the MD-LEVEL and pass the packet to the Opcode de-multiplexer. The Opcode de-multiplexer delivers CCM packets to the CCM process.

The CCM Process performs processing specified in [8021Q].

Additionally the CCM process updates the CCM Management Object with the sequence number of the received CCM packet. Note: The last received CCM sequence number and CCM timeout are tracked per each remote MEP.

If the CCM timeout is true for the sending remote MEP, then clear the CCM timeout in the CCM Management object and generate the SNMP notification as specified above.

## 12. Security Considerations

Specific security considerations related methods presented in this document are currently under investigation.

## 13. IANA Considerations

IANA is requested to allocate 4 CFM Op-codes and 10 CFM TLV types and enter them into the IANA "CFM OAM IETF Op-codes" and "CFM OAM IETF TLV Types" sub-reigistries:

TBD values

## 14. References

### 14.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [8021Q] IEEE, "Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks", IEEE Std 802.1Q-2011, August, 2011.

### 14.2. Informative References

- [RFC4379] Kompella, K. et.al, "Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures", RFC 4379, February 2006.
- [RFC6291] Andersson, L., et.al., "Guidelines for the use of the "OAM" Acronym in the IETF" RFC 6291, June 2011.
- [Y1731] ITU, "OAM functions and mechanisms for Ethernet based networks", ITU-T G.8013/Y.1731, July, 2011.
- [NVO3FRM] Lasserre, M., et.al., "Framework for DC Network Virtualization", Work in Progress, January, 2014.
- [NVO3ARC] Black, D., et.al., "Architecture for Overlay Networks (NVO3)", Work in Progress, December, 2013.
- [NVO3OAMREQ] Ashwood-Smith, P., "NVO3 Operations, Administrations and Maintenance Requirements", work in progress, January, 2014.
- [NVO3DPREQ] Bitar, N., "NVO3 Data Plane Requirements", Work in Progress, November, 2013.

## 15. Acknowledgments

This document was prepared using 2-Word-v2.0.template.dot.

## Appendix A.

## Base Mode for NVO3 OAM

CFM, as defined in [8021Q], requires configuration of several parameters before the protocol can be used. These parameters include MAID, Maintenance Domain Level (MD-LEVEL) and MEPIDs. The Base Mode for NVO3 OAM defined here facilitates ease of use and provides out of the box plug-and-play capabilities.

All NVO3 compliant devices that support OAM specified in this document MUST support Base Mode operation.

All NVO3 compliant devices that support OAM MUST create a default MA with MAID as specified herein.

MAID [8021Q] has a flexible format and includes two parts: Maintenance Domain Name and Short MA name. In the Based Mode of operation, the value of the Maintenance Domain Name must be the character string "NVO3BaseMode" (excluding the quotes "). In Base Mode operation Short MA Name format is set to 2-octet integer format (value 3 in Short MA Format field) and Short MA name set to 65532 (0xFFFFC).

The Default MA belongs to MD-LEVEL 3.

In the Base Mode of operation, each NVO3 device creates a single UP MEP associated with a virtual OAM port located within the CPU with no physical layer (NULL PHY). The MEPID associated with this MEP is the 2-octet VNI Nickname.

By default, with the base mode OAM in place, all NVO3 devices operating in the Base Mode for NVO3 OAM are able to initiate LBM, PT and other OAM tools with no configuration.

Implementation MAY provide default flow to be included in OAM messages. Content of the default flow is outside the scope of this document.

Figure 18, below depicts encoding of MAID within the CCM messages.

Field Name	Size
Maintenance Domain Format	1
Maintenance Domain Length	2
Maintenance Domain Name	variable
Short MA Name Format	1
Short MA Name Length	2
Short MA Name	variable
Padding	Variable

Figure 18 MAID structure as defined in [8021Q]

Maintenance Domain Name Format is set to Value: 4

Maintenance Domain Name Length is set to value: 13

Maintenance Domain Name is set to: NVO3BaseMode

Short MA Name Format is set to value: 3

Short MA Name Length is set to value: 2

Short MA Name is set to : FFFC

Padding : set of zero up to 48 octets of total length of the MAID.

Please refer to [8021Q] for details.

Authors' Addresses

Tissa Senevirathne  
CISCO Systems  
375 East Tasman Drive.  
San Jose, CA 95134  
USA.

Phone: +1 408-853-2291  
Email: tsenevir@cisco.com

Norman Finn  
CISCO Systems  
510 McCarthy Blvd  
Milpitas, CA 95035  
USA

Email: nfinn@cisco.com

Samer Salam  
CISCO Systems  
595 Burrard St. Suite 2123  
Vancouver, BC V7X 1J1, Canada

Email: ssalam@cisco.com

Deepak Kumar  
CISCO Systems  
510 McCarthy Blvd,  
Milpitas, CA 95035, USA

Phone : +1 408-853-9760  
Email: dekumar@cisco.com

Donald Eastlake  
Huawei Technologies  
155 Beaver Street  
Milford, MA 01757

Phone: +1-508-333-2270  
Email: d3e3e3@gmail.com

Sam Aldrin  
Huawei Technologies  
2330 Central Express Way  
Santa Clara, CA 95951  
USA

Email: [aldrin.ietf@gmail.com](mailto:aldrin.ietf@gmail.com)





Network working group  
Internet Draft  
Category: Standard Track

L. Xia  
L. Yong  
Huawei

Expires: September 2014

February 14, 2014

Layer 2 Gateway (L2GW)  
draft-xia-nvo3-l2gw-00

## Abstract

Layer 2 Gateway (L2GW) is used for interconnecting layer 2 overlay network [NVO3FRWK] with layer 2 bridge networks [IEEE802.1Q] to form one layer 2 virtual network. This draft discusses which Layer 2 Control Protocol (L2CP) specified in IEEE802.1 should be supported by the layer 2 overlay network and which not, and specifies how L2GW should deal with L2CP frames.

## Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at  
<http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at  
<http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on September 14, 2014.

## Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

## Table of Contents

1. Introduction .....	3
1.1. Conventions used in this document .....	4
1.2. Terminology .....	4
2. L2GW Reference Model.....	4
3. L2CP Review and Applicability to L2 Overlay Network.....	5
3.1. STP/RSTP/MSTP .....	7
3.2. PAUSE .....	7
3.3. LACP/LAMP .....	7
3.4. Link OAM .....	8
3.5. Port Authentication .....	9
3.6. E-LMI .....	9
3.7. LLDP .....	9
3.8. PTP Peer Delay .....	10
3.9. ESMC .....	10
3.10. GARP/MRP Block.....	10
4. L2CP Process in L2GW.....	10
4.1. L2CP Frames Filtered (Peered or Discarded) in L2GW ....	11
4.2. L2CP Frames Passed through L2GW .....	11
5. Other Interworking Cases .....	12
6. Security Considerations .....	12
7. IANA Considerations .....	12
8. References .....	12
8.1. Normative References .....	12
8.2. Informative References .....	13

## 1. Introduction

Cloud computing and network virtualization is evolving to the direction of virtualized networks in overlay, which aims fast and easy to create tenant networks, support tenant system mobility, and bring the manageability of all virtualized resources in DC.

Layer 2 (L2) overlay network in NVO3 means an L2 overlay network interconnecting tenant systems, where any pair of Network Virtualization Edges (NVE) are connected by IP tunnels. As a result, it forms a full mesh topology of overlay network, i.e. only one hop between any pair of NVEs. L2 bridge network in this draft is the network specified in IEEE 802.1Q [IEEE 802.1Q] which are widely deployed in DCs. L2 overlay network is used to refer the L2 network specified in NVO3.

During DC network migration, it's very common that L2 overlay network may be mix used with L2 bridge network in a DC, and the communication between them is required. In another use case, using L2 bridge network to connect non-virtualized devices in DC are mature and well deployed method in DC; these non-virtualized devices are necessary for some applications such as BIG data and are required to communicate with virtualized resources.

To interconnect an L2 overlay network with an L2 bridge network, gateway functions are needed on the device(s)/system(s) connecting two networks. This device is referred as to Layer 2 Gateway (L2GW) in this draft. L2GW processes the encapsulation translation of packets between overlay technologies (e.g., VxLAN [VXLAN], NVGRE [NVGRE]) and the technologies specified in IEEE 802.1Q; it also deals with Layer 2 Control Protocol (L2CP) frames from the bridge network.

L2CP frames are defined by IEEE802.1 to be used for L2 network control, e.g., STP, LACP, etc. An L2CP is identified by one of the following MAC destination addresses:

- o 01-80-C2-00-00-00 through 01-80-C2-00-00-0F: Bridge Block of protocols
- o 01-80-C2-00-00-20 through 01-80-C2-00-00-2F: GARP/MRP Block of protocols

All L2CPs are supported in a L2 bridge network, every 802.1Q bridge performs corresponding action, i.e., peer (to be processed by local protocol entity), discard, pass, based on L2CP frame's MAC destination address (DA) and protocol identifier (Ethertype or LLC

Address). For L2 overlay VN, most of L2CPs are unnecessary because L2 overlay VN has its own control plane functions to support needed L2 communication such as transport over a tunnel or OAM. It is very useful to document how these service frames should be handled at L2GW to ensure that two networks can interwork.

This draft discusses which L2CP should be supported by L2 overlay network and which not, and specifies how L2GW should deal with L2CP frames.

### 1.1. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RFC2119].

### 1.2. Terminology

This document uses the terms defined in NVO3 framework [NVO3FRWK] and architecture [NVO3ARCH] documents.

## 2. L2GW Reference Model

The following diagram shows a reference model where L2GW provides an interconnection between L2 overlay network and L2 bridge network.

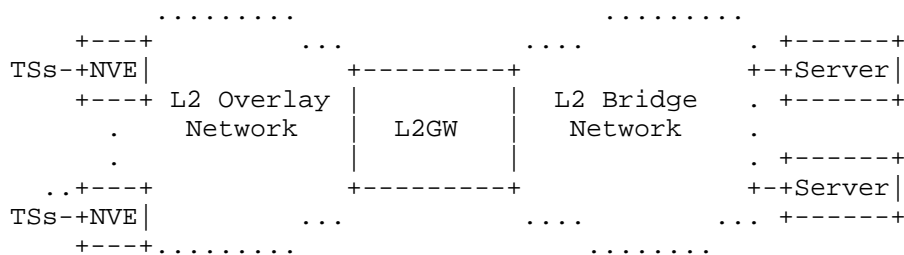


Figure 1: L2GW Reference Model

L2GW can reside on access switch providing direct connection with physical server, or reside on core switch or edge router providing connection with L2 bridge network.

Note that this draft only addresses the case that L2GW and L2 NVE are co-located at the edge device between two L2 networks, where L2GW can deal with the interworking work. Other cases are not yet covered in this draft.

## 3. L2CP Review and Applicability to L2 Overlay Network

L2CP protocols defined in IEEE 802.1 are listed in Table 1:

MAC DA	Assignment	Protocol Type	L2CP Action	
			VLAN-based L2 services	PORT-based L2 services
01-80-C2-00-00-00	Nearest Customer Bridge	STP/RSTP/MSTP, LACP/LAMP	Filter	Pass
01-80-C2-00-00-01	IEEE MAC Specific Control Protocols	PAUSE	Filter	Filter
01-80-C2-00-00-02	IEEE 802 Slow Protocols	LACP/LAMP, Link OAM, ESMC	Filter	Filter
01-80-C2-00-00-03	Nearest non-TPRM Bridge	Port Authentication, LACP/LAMP	Filter	Filter
01-80-C2-00-00-04	IEEE MAC Specific Control Protocols		Filter	Filter
01-80-C2-00-00-05	Reserved for Future Standardization		Filter	Filter
01-80-C2-00-00-06				
01-80-C2-00-00-09				
01-80-C2-00-00-0A				

01-80-C2-00-00-07	MEF ELMI	E-LMI	Filter	Filter
01-80-C2-00-00-08	Provide Bridge Group		Filter	Filter
01-80-C2-00-00-0B	Reserved for Future Standardization		Filter	Pass
01-80-C2-00-00-0C				
01-80-C2-00-00-0D	Provider Bridge MVRP		Filter	Pass
01-80-C2-00-00-0E	Nearest Bridge, Individual LAN Scope	LLDP, PTP Peer Delay	Filter	Filter
01-80-C2-00-00-20		GARP/MRP Block	Pass	Pass
through				
01-80-C2-00-00-2F				

Table 1 L2CP protocols specification

## Note:

Different L2CP protocols can use the same MAC DA in above block of 32 addresses, but be differentiated by protocol identifier. MAC DA determines the intended recipient device for the frame;

Filter represent the L2CP action of peer or discard;

Based on whether L2 interface is VLAN-aware, L2 services can divided into two categories: VLAN-based L2 services, PORT-based L2 services. L2CP action (peer, discard, pass) for these two L2 services is also different;

Whether the L2CP frames are peered or discarded is further determined by the configuration of L2 interface.

Further analysis about whether a L2CP protocol is necessary and how it is processed in NVO3 supported L2 VN, is provided in the following sub sections.

### 3.1. STP/RSTP/MSTP

The Spanning Tree Protocol (STP) is a L2 protocol that ensures a loop-free topology for any bridged Ethernet local area network. The basic function of STP is to prevent bridge loops and the broadcast storm that results from them. Rapid spanning Tree Protocol (RSTP) and Multiple Spanning Tree Protocol (MSTP) are all the enhanced xSTP protocols.

L2 overlay network does not need xSTP protocols to prevent bridge loops because it has its own mechanism for it, i.e., NVA, control plane mechanisms, full mesh + split horizon, etc. So, the process of xSTP frames in L2 VN is:

Be in line with L2CP protocols' specification of Table 1 from IEEE in the L2 sub-networks attached to L2 NVEs;

xSTP frames are filtered in L2 NVEs and should not go into L2 overlay network.

### 3.2. PAUSE

[IEEE 802.3-2005] has specified a L2 flow control mechanism through using the PAUSE frame. This frame uses L2CP MAC DA of 01-80-C2-00-00-01 to be sent to the node at the other end of the link for informing it to halt the frame transmission for a specified period of time.

When L2 NVE is co-located in Hypervisor, PAUSE frame is not necessary in one device. When they are separated, PAUSE frame is only used in layer 2 network between L2 NVE and Hypervisor, there is no need to overlay PAUSE frame between L2 NVEs.

### 3.3. LACP/LAMP

Link Aggregation [IEEE 802.1AXbk-2012] is a mechanism for making multiple point-to-point links between a pair of devices appear to be a single logical link between those devices. Link Aggregation

Control Protocol (LACP) and Link Marker Control Protocol (LAMP) operate between exactly two peer devices for the purpose of creating, verifying, and monitoring the logical link created by aggregating individual links. Specific L2CP frames, known as Link Aggregation Control Protocol Data Units (LACPDUs), are exchanged between the peer devices on each individual link in the aggregation. The protocol identifier used by LACP is an Ethertype with a value of 0x8809 (the ''Slow Protocols'' Ethertype) and subtype values 01 (for LACP) and 02 (for LAMP). Note that LACP is used to represent LACP and LAMP in the following text.

LACP uses 3 different L2CP MAC DAs to determine the scope of propagation of LACPDUs within a bridged LAN, as Table 2 follows:

Assignment	L2CP MAC DA	Peered or discarded by
Nearest Customer Bridge	01-80-C2-00-00-00	End Station, Customer Bridge, Provider Edge Bridge
IEEE 802 Slow Protocols	01-80-C2-00-00-02	End Station, Customer Bridge, Provider Edge Bridge, Provider Bridge
Nearest non-TPRM Bridge	01-80-C2-00-00-03	Bridges except for Two Port MAC Relay

Table 2 LACP specification of L2CP MAC DAs

Base on the summary of Table 2, LACPDUs with the L2CP MAC DA of 01-80-C2-00-00-02 are peered or discarded by every node, so this kind of LACPDUs will not be overlaid across the L2 overlay network. For 01-80-C2-00-00-00, it is possible that LACPDUs need to be overlaid across Provider Bridge and L2 NVEs of L2 overlay network to reach the other end Custom Bridge, L2 overlay network maybe need to support to overlay this kind of LACP frame between L2 NVEs. How the L2 overlay network support LACP frame of 01-80-C2-00-00-03 is TBD.

### 3.4. Link OAM

Lin OAM defined is defined in [IEEE 802.3ah], as mechanisms for monitoring and troubleshooting Ethernet access links. Specifically it defines tools for discovery, remote failure indication, remote and local loopbacks and status and performance monitoring.



The Link OAM frames using L2CP MAC DA of 01-80-C2-00-00-02 are peered or discarded by every node, so this kind of frame will not be overlaid across the L2 overlay network.

### 3.5. Port Authentication

[IEEE 802.1X] is an IEEE Standard for Port-based Network Access Control (PNAC). It is part of the IEEE 802.1 group of networking protocols. It provides an authentication mechanism to devices wishing to attach to a LAN or WLAN.

Whether or not the L2 overlay network needs to overlay this L2CP frames is TBD.

### 3.6. E-LMI

Ethernet Local Management Interface (E-LMI) is a protocol between the customer edge (CE) device and the provider edge (PE) device. It runs only on the PE-CE UNI link and notifies the CE of connectivity status and configuration parameters of Ethernet services available on the CE port. E-LMI interoperates with an OAM protocol, such as Connectivity Fault Management (CFM), that runs within the provider network to collect OAM status. CFM runs at the provider maintenance level (UPE to UPE with inward-facing MEPs at the UNI). E-LMI relies on the OAM Ethernet Infrastructure (EI) to interwork with CFM for end-to-end status of Ethernet virtual connections (EVCs) across CFM domains.

The LLDP frames using L2CP MAC DA of 01-80-C2-00-00-07 are peered or discarded by every node except for the Two Port MAC Relay (TPMR) bridge, so this kind of frame will not be overlaid across the L2 overlay network.

### 3.7. LLDP

The Link Layer Discovery Protocol (LLDP) is a vendor-neutral link layer protocol in the Internet Protocol Suite used by network devices for advertising their identity, capabilities, and neighbors on an IEEE 802 local area network, principally wired Ethernet. The protocol is formally referred to by the IEEE as Station and Media Access Control Connectivity Discovery specified in standards document [IEEE 802.1AB].

The LLDP frames using L2CP MAC DA of 01-80-C2-00-00-0E are peered or discarded by every node, so this kind of frame will not be overlaid across the L2 overlay network.

### 3.8. PTP Peer Delay

PTP Peer Delay frame is specified in [IEEE 1588-2008] to carry PTP peer time information. It uses L2CP MAC DA of 01-80-C2-00-00-0E and peered or discarded by every node, so this kind of frame will not be overlaid across the L2 overlay network.

### 3.9. ESMC

Ethernet Synchronization Messaging Channel (ESMC) is specified in [ITU-T Rec. G.8264] for conveying clock information between Synchronous Ethernet (SyncE) bridges.

The ESMC frames using L2CP MAC DA of 01-80-C2-00-00-02 are peered or discarded by every node, so this kind of frame will not be overlaid across the L2 overlay network.

### 3.10. GARP/MRP Block

Multiple Registration Protocol (MRP), which replaced Generic Attribute Registration Protocol (GARP), is a generic registration framework defined by the [IEEE 802.1ak] amendment to the [IEEE 802.1Q] standard. MRP allows bridges, switches or other similar devices to be able to register and de-register attribute values, such as VLAN identifiers and multicast group membership across a large LAN. MRP operates at the Data Link Layer.

The block of L2CP MAC DA from 01-80-C2-00-00-20 to 01-80-C2-00-00-2F is used for MRP protocol. Now, only 01-80-C2-00-00-20 is for Multiple MAC Registration Protocol (MMRP) and 01-80-C2-00-00-21 is for Multiple VLAN Registration Protocol (MVRP), other L2CP MAC DA of the block are all reserved for future use. Protocol use one address of this block is passed by all the intervening bridges that does not participate in the protocol using this address, and peered or discarded by the bridge that participate in the protocol at last. This forwarding rule maybe requires MRP frames to be overlaid across the L2 overlay network.

## 4. L2CP Process in L2GW

For all L2CP protocols, several differences exist between L2 overlay network and L2 bridge network on how to process them. As the demarcation point between L2 overlay network and L2 bridge network, L2GW keeps the same action to all L2CP frames as before at the L2 bridge network side on the one hand, but maybe processes some L2CP frames differently at the L2 overlay network side on the other hand. The following sub sections will describe the L2CP process in L2GW.

#### 4.1. L2CP Frames Filtered (Peered or Discarded) in L2GW

Although xSTP protocols using Nearest Customer Bridge address of 01-80-C2-00-00-00 indicate that it can be overlaid across L2 overlay network, they still are not necessary for L2 overlay network because L2 overlay network has its own mechanism to prevent bridge loops. So xSTP frames will be filtered by the L2GW and not go into the L2 overlay network.

Based on the analysis of section 3.3, LACP/LAMP frames using IEEE 802 Slow Protocols of 01-80-C2-00-00-02 are not necessary for L2 overlay network. So, LACP/LAMP frames will be filtered by the L2GW and not go into the L2 overlay network. ESMC frames using the same MAC DA will also be filtered by L2GW.

For Link OAM frames, if OAM functions are necessary for the whole L2 network which interconnects L2 bridge network and L2 overlay network, L2GW needs to support the interworking of OAM as well. This means that L2GW should peer the Link OAM frames of L2 bridge network and perform some actions between NVEs in L2 overlay network. The detailed operation is TBD.

Other L2CP protocols that are filtered by L2GW and do not go into L2 overlay network include PAUSE, E-LMI, LLDP, PTP Peer Delay. The basic reason is that they all require to be processed hop by hop in L2 network strictly, but overlay network breaks this rule.

The action of "filter" can be "peer", or "discard". It depends on the specific service requirement, i.e., does L2GW need to participate in the L2CP protocol, etc. How to determine the specific action is TBD.

#### 4.2. L2CP Frames Passed through L2GW

Excepting for the aforementioned L2CP protocols filtered by L2GW, the left L2CP protocols need to be passed through L2GW. They include:

- LACP/LAMP frames using IEEE 802 Slow Protocols of 01-80-C2-00-00-00;

- GARP/MRP series protocols (i.e., MMRP, MVRP) using the MAC DA block of 01-80-C2-00-00-20 through 01-80-C2-00-00-2F.

All these kinds of L2CP frames are passed through L2GW and traverse across the L2 overlay network and L2 bridge network to arrive the bridges that participate in the L2CP protocols. For MRP protocols, another necessary operation of L2GW is to use the pre-provisioned

VLAN to virtual network instance (VNI) mappings in NVE locally or by getting from NVA to map these MRP frames into corresponding VNIs.

## 5. Other Interworking Cases

There are other L2 bridge network technologies that use L2 Control Plane protocols such as Provider Bridge [IEEE802.1AD] or Provider Backbone Bridge [PBB] [IEEE802.1AH]. The use case of L2 Overlay Network interworking with these types of bridge networks is for the further study.

Note that VPLS [RFC4761] [RFC4762], EVPN [EVPN], Shortest Path Bridging [IEEE SPB] and TRILL [RFC6325] are also technologies for L2 private network implementation. These technologies rely on the control plane protocol and aim for service provider network. SDN controller interworking with such control plane protocol will be addressed in separate draft.

## 6. Security Considerations

TBD.

## 7. IANA Considerations

The document does not require any IANA action.

## 8. References

### 8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC2119, March 1997.

[RFC4761] Kompella, K. and Rekhter, Y. (Editors), "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling", RFC 4761, January 2007

[RFC4762] Lasserre, M. and Kompella, V. (Editors), "Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling", RFC 4762, January 2007.

[RFC6325] Perlman, R., "Rbridges: Base Protocol Specification", July 2011.

## 8.2. Informative References

[NVO3ARCH] Black, D, Narten, T., et al, "An Architecture for Overlay Networks (NVO3)", draft-narten-nvo3-arch-01, work in progress

[NVO3FRWK] LASSERRE, M., Motin, T., et al, "Framework for DC Network Virtualization", draft-ietf-nvo3-framework-03, work in progress.

[NVGRE] Sridharan, M., et al, "NVGRE: Network Virtualization using Generic Routing Encapsulation", draft-sridharan-virtualization-nvgre-03, work in progress

[VXLAN] Mahalingam, M., Dutt, D., etc, "VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", draft-mahalingam-dutt-dcops-vxlan-05.txt, work in progress

[EVPN] Sajassi, A. and R. Aggarwal, "BGP MPLS Based Ethernet VPN", draft-ietf-l2vpn-evpn-04, July 2013

[IEEE 802.1Q] "Virtual Bridged Local Area Networks", 2005

[IEEE 802.3-2005] "Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications"

[IEEE 802.1AXbk-2012] "IEEE Standard for Local and metropolitan area networks--Link Aggregation Amendment 1: Protocol Addressing"

[IEEE 802.3ah] "IEEE Standard for Information technology--Local and metropolitan area networks--Part 3: CSMA/CD Access Method and Physical Layer Specifications Amendment: Media Access Control Parameters, Physical Layers, and Management Parameters for Subscriber Access Networks"

[IEEE 802.1X] "IEEE Standard for Local and metropolitan Area Networks. Port-based Network Access Control"

[IEEE 802.1AB] "IEEE Standard for Station and Media Access Control, Connectivity Discovery"

[IEEE 1588-2008] "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems"

[IEEE 802.1ak] "IEEE Standard for Local and metropolitan Area Networks - Virtual Bridged Local Area Networks, Amendment 7: Multiple Registration Protocol"

[IEEE 802.1AD], "Virtual Bridged Local Area Networks - Amendment 4: Provider Bridges", 2005

[PBB] Clauses 25 and 26 of "IEEE Standard for Local and metropolitan area networks - Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks", IEEE Std 802.1Q, 2013.

[IEEE802.1AH] IEEE Draft P802.1ah/D4.2 "Virtual Bridged Local Area Networks, Amendment 6: Provider Backbone Bridges", 2008

[IEEE SPB] "IEEE standard for local and metropolitan area networks: Media access control (MAC) bridges and virtual bridged local area networks -- Amendment 20: Shortest path bridging", IEEE 802.1aq, June 2012.

[ITU-T Rec. G.8264] "Distribution of Timing Through Packet Networks"

#### Authors' Addresses

Liang Xia  
Huawei Technologies

Email: frank.xialiang@huawei.com

Lucy Yong  
Huawei Technologies, USA

Email: lucy.yong@huawei.com



Network working group  
Internet Draft  
Category: Standard Track

L. Yong  
L. Xia  
Huawei  
Q. Zu  
Ericsson

Expires: September 2014

February 13, 2014

Network Virtualization Edge (NVE)  
draft-yong-nvo3-nve-03

Abstract

This document specifies Network Virtualization Edge (NVE) data plane interoperability functionality for Network Virtualization Overlays (NVO3). These specifications are necessary for the interoperability between an NVE and its attached tenant systems and between the NVEs.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on September 13, 2014.



## Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

## Table of Contents

1. Introduction.....	3
1.1. Conventions used in this document.....	3
1.2. Terminology.....	3
2. NVE Design Principles.....	3
3. Tenant Systems.....	4
3.1. Virtual Machines and Bare Metal Servers.....	4
3.2. Network Service Appliances.....	5
3.3. Gateways.....	5
4. Network Virtualization Edge (NVE).....	6
4.1. NVE Service Type.....	6
4.1.1. L2 NVE Service.....	6
4.1.2. L3 NVE Service.....	7
4.1.3. L2/L3 NVE Service.....	9
4.2. Overlay Tunnel between NVEs.....	11
4.3. Multi-Tenancy Support.....	12
4.4. Route Path Control.....	12
4.5. Split-NVE.....	13
4.6. Multi-Homing Support.....	13
4.7. OAM Tools on NVE.....	14
5. Operation Considerations.....	15
5.1. VM Mobility.....	15
5.2. Gateway vs. Distributed Gateway.....	15
6. Security Considerations.....	18
7. Acknowledgements.....	19
8. IANA Considerations.....	19
9. References.....	19
9.1. Normative References.....	19
9.2. Informative References.....	19

## 1. Introduction

Network Virtualization Edge (NVE) is a component in Network Virtualization Overlays Technology. This component is described in the NVO3 framework [NVO3FRWK] and architecture [NVO3ARCH]. This document specifies NVE data plane functionality. The functionality specifications are necessary for the interoperability between an NVE and its attached tenant systems and between the NVEs. The data plane functionality described in this document is independent of NVO3 control plane functionality. Thus, the control plane functionality is outside the scope of this document. However the specifications in this document can support any control plane implementation and are helpful control plane protocol development.

NVE data plane functionality essential is the packet forwarding. It receives a packet from a tenant system via a virtual access point (VAP), processes it, and forwards it to the peer NVE via an overlay tunnel or forwards to a local VAP; it receives a packet from a peer NVE, processes it, and forwards it to a tenant system via a VAP. In the process, an NVE may modify the packet header and/or insert/remove the tunnel header on the packet prior to the forwarding. This document does not specify tunnel encapsulation protocol but describe the usage at NVE.

In order to make NVO3 data plane work properly, some configurations on NVEs are necessary. They can be done manually or automated. How these configurations are done is outside the scope of the document.

### 1.1. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RFC2119].

### 1.2. Terminology

This document uses the terms defined in NVO3 framework [NVO3FRWK] and architecture [NVO3ARCH] documents.

## 2. NVE Design Principles

NVE design principles are:

1. The solution supports multi-tenancy in a common underlying network.

2. The solution supports different types of tenant systems and requires no change on tenant system configuration and behavior.
3. The solution is agnostic to NVE location, i.e. regardless NVE is co-located with tenant systems on a server/device or physically separated from tenant systems.
4. No change on tenant system configuration and behavior whether a gateway or distributed gateway is used.
5. The solution must support virtual machine (VM) mobility.
6. The solution must be scalable in supporting a VN having many NVEs each of which may have many attached tenant systems; and in supporting an NVE being the members of several VNs and having attached tenant systems that belong to the same or different VNs.

Note that NVO3 architecture [NVO3ARCH] defines NVE and NVA entities; item 5 and 6 achievement depends on both NVE and NVA. This document only focuses on NVE data plane functionality. The interaction between NVE and NVA, between NVAs, and between hypervisor and NVE are outside the scope of the document.

### 3. Tenant Systems

NVO3 architecture [NVO3ARCH] defines several types of tenant systems. Following sections describes these tenant systems in terms of their role and networking behavior.

#### 3.1. Virtual Machines and Bare Metal Servers

Tenant system may be a virtual machine on a server or a bare metal server. For a virtual machine, Guest OS runs on the tenant system and application software runs on top of Guest OS. For a bare metal server, host OS runs on the server and application software runs on top of it. Here is the summary of such tenant system networking behavior:

- . A tenant system (TS) is configured with a subnet such as 10.1.1.0/24 and a default GW IP address, and TS MAC and IP address (manually or automatically). Note that TS IP address MUST be an address in the subnet. How to configure these addresses is outside the scope of document.

- . A tenant system learns the MAC address of the peer in the same subnet by using a protocol (e.g. ARP, NDP) and may learn it from the source MAC address on the packet as well.
- . A tenant system learns the GW MAC address from ARP or NDP protocol. The GW entity MUST support ARP protocol.
- . A tenant system may cache the interested destination IP and MAC address for the packet forwarding.
- . For intra subnet forwarding, a tenant system inserts the destination MAC address on the packet.
- . For inter subnet forwarding, a tenant system inserts the GW MAC address on the packet.
- . A tenant system may filter received packets and only accept the packet with the designation MAC address the same as its MAC address.

### 3.2. Network Service Appliances

A network service appliance such as load balancer or firewall can act as a tenant system and provide a service to one or more VNs (via distinct VAPs). A network service appliance can be implemented on a physical device, a bare metal server, or a virtual machine on a server. The last one is often referred as Virtualized Network Function (VNF). A tenant system, as a network service appliance, may have different configuration and behavior as a host as described in section 3.1. The tenant system may attach to an NVE via a local VLAN interface, IP interface, or directly attached port/vport and act as a middle box in a VN. Typically, the configuration or policy on the local NVE determines which VN traffic or traffic flows are forwarded to the tenant system. In other words, the local NVE does not forward the packets to the tenant system based on the destination address on the packets. Such tenant system may also be the member of multiple VNs via distinct virtual access points (VAPs), respectively, and may forward the received traffic back to the same VN or to a different VN. The tenant system may even modify the received packets prior to forwarding them, e.g. Network Address Translation (NAT).

### 3.3. Gateways

A gateway may be used to interconnect two VNs implemented by NVO3 (refer it as to NVO3 VN), between an NVO3 VN and other networks that may be virtual, physical, between an NVO3 VN and Internet, or a combination of these. A gateway may also interconnect two NVO3 VNs

that are implemented with different NVE service types. A gateway may be implemented on a physical device, a bare metal server, or a VM.

Note that a distributed gateway may be implemented for NVO3 VNs interworking. The distributed gateway means that a gateway function is implemented on NVEs so that the traffic between the VNs can be forwarded at the local NVE directly; as a result path optimization is gained. This document describes the distributed gateway function on NVEs.

It is often that a gateway integrates with several other network service appliances such as NAT, firewall and policy based forwarding for the interconnection need, which means that inter-VN traffic gets these special treatments.

A gateway can be implemented as a Tenant System, in which it is like a network service appliance described in section 3.2. A gateway can also be implemented on a network device, i.e. have embedded NVE component; which means that the device supports NVE capability and is able to interwork with other NVEs seamlessly. In the rest of document, it describes a gateway as the second case. The first case is categorized as a network service appliance.

#### 4. Network Virtualization Edge (NVE)

NVO3 framework [NVO3FRWK] defines three NVE service types: L2 service, L3 service, and L2/L3 service. A tenant network can be implemented with one of NVE types or the combination w/ a gateway.

Note the document uses ARP protocol to describe interoperability function between NVE and Tenant System. The use of NDP protocol is for next version.

##### 4.1. NVE Service Type

###### 4.1.1. L2 NVE Service

An L2 VN is implemented with L2 NVE service type and provides L2 broadcast domain to the tenant systems on the VN. The tenant system attaches to the NVE via a VLAN, directly attached port or virtual port. For the scalability and performance concern, an NVE should not forward the ARP request message to remote NVEs. If the interested tenant system at a remote NVE, the local NVE sends the ARP response with the remote MAC address back. An NVE can obtain the remote tenant system MAC address via NVA.[NVO3ARCH] The L2 overlay is used between NVEs (NVE gets the inner/outer address mapping from NVA). Note that, an L2 virtual network overlay provides a single broadcast

domain. Figure 1 illustrates a VN with L2 NVE service type. As shown in the Figure, TSs attach to an NVE via the VAPs that are associated to the L2 VNix on the NVE. L2 Overlay is used between the NVEs that are the members of a VN. The VN ID is encoded in the overlay header on the packets.

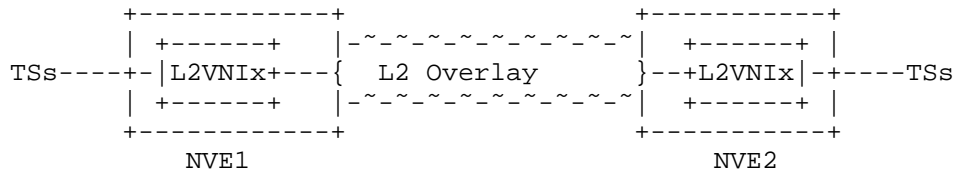


Figure 1 L2 NVE Service Model

L2 NVE service supports a broadcast domain in the VN. How to transport broadcast/multicast traffic among NVEs requires a way to map the VN broadcast/multicast traffic to a underlying IP multicast solution. IP network does not support a multicast solution yet and relies on Protocol Independent Multicast (PIM) to support multicast transport. An NVE can send the VN broadcast/multicast traffic to the remote NVEs by using unicast outer IP address on the packets, i.e. replicating in the underlying network. This method does not require the underlying network to support a multicast transport. How NVA conveys such mapping information to NVEs is outside the scope of the document.

To interconnect with external virtual or physical networks or an overlay or non-overlay virtual network, a gateway is necessary. The gateway participates in the L2 VN and performs the traffic treatment between the interconnected networks.

L2 NVE service may apply to non-IP and IP applications. However IP based application can be implemented in other ways too. (see below)

To use NVE data plane learning of mapping between remote tenant system MAC and remote NVE IP address mapping, and for a remote tenant system to use ARP in notifying its MAC address is for further study.

#### 4.1.2. L3 NVE Service

An L3 VN is implemented with L3 NVE service type and provide L3 routing domain for the tenant systems in the VN. The VAP has an IP interface, attached port/vport, or a VLAN interface. An NVE acts as the first hop router to the attached tenant systems. For the VLAN interface, i.e. Ethernet access interface, the NVE needs support ARP

protocol, terminates all ARP request messages, and reply its MAC address to the tenant system. An NVE tracks the tenant system IP and MAC address mapping if L2 access is used unless the special configuration is done on the NVE. (see section 3.2).

When a tenant system forwards packets to an attached NVE, the NVE receives either IP packets or Ethernet frames with NVE MAC address as the destination MAC on the packets depending on VAP interface type. The NVE performs an IP table lookup based on the destination IP address on the packets. If the packet needs to forward to another NVE, the NVE sends it via L3 overlay (NVE obtains the inner/outer address mapping from NVA). If the packet needs to be forwarded to a local VAP, and the VAP is Ethernet access, the NVE inserts the destination MAC address on the packet prior to sending it to the tenant system; if the VAP is an IP interface, the NVE sends IP packet directly.

How NVE obtains local tenant system IP and MAC addresses is outside the scope of this document. The rule of thumb is that such method MUST not require any change on the tenant system side.

The tenant systems connecting to an L3 VN can be on the same or different subnets. A special case of this VN is that the VN provides the connectivity for all intra and inter subnets, which brings optimal path. Typically, subnet or flow based policies may be configured for route constraint or route path control policies may be configured depending on the tenant network requirements. Figure 2 depicts an L3 VN model.

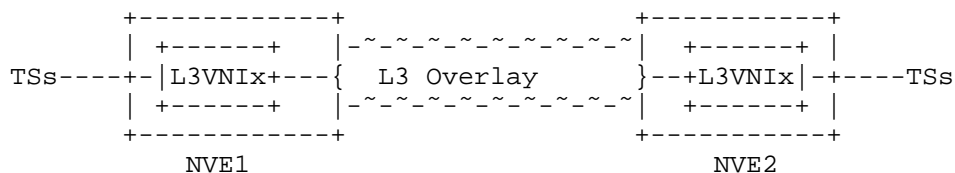


Figure 2 L3 NVE Service Model

If an L3 VN needs to interconnect with external networks, Internet, or another VN, a gateway MUST be used. To avoid address collision, either IP address space partition or IP address translation between two VNs at a gateway is necessary. The former, in turn, looks like one routing domain with one IP space shared between two virtual networks. For two L3 VNs interconnection, NVEs may be function as distributed gateway and perform both intra-VN routing/forwarding and

inter-VN routing/forwarding/gateway. Figure 3 depicts an L3 NVE w/ distributed gateway. As shown in the Figure, TSSs attach to an NVE via the VAPs that are associated to the L3 VNix or L3 VNiy on the NVE. L3 Overlay is used between the NVEs that are the members of the VNx and VNy. The VNx and VNy IDs are encoded in the overlay header on the packets. Section 5.3 discusses operation considerations in use of a gateway or distributed gateway.

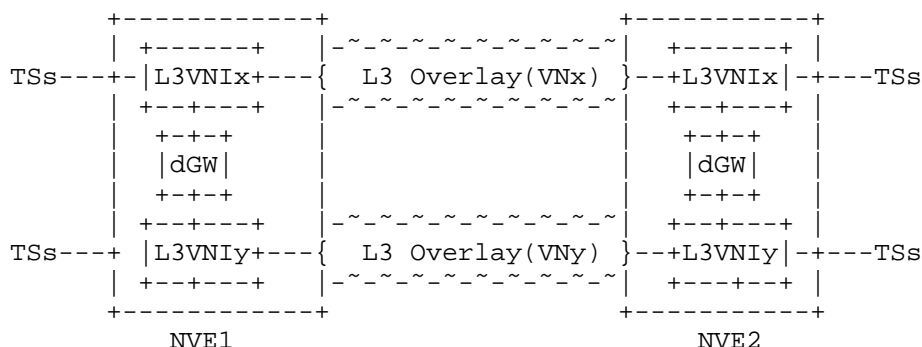


Figure 3 L3 NVE Service w/dGW Model

Note: an L3 VN, as a route domain, does not support broadcast and multicast function. Applicability for MVPN [RFC4834] to NVO3 is for further study. It is obvious that L3 NVE service type only supports IP based application.

### 4.1.3. L2/L3 NVE Service

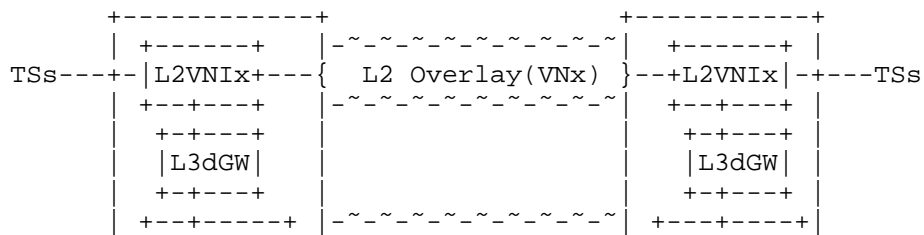
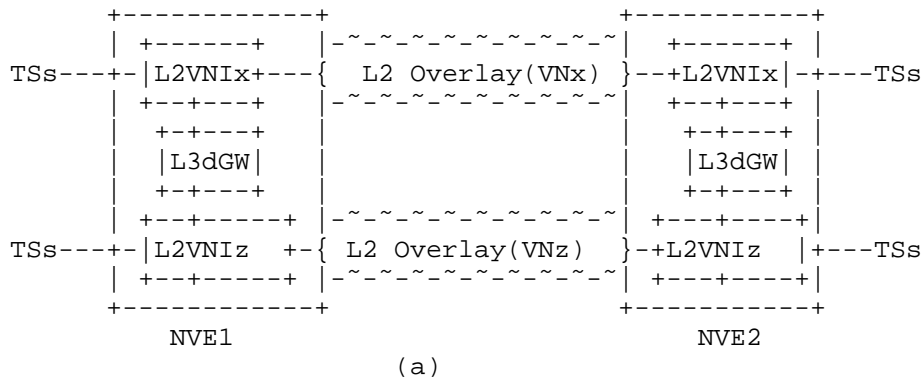
L2/L3 NVE service type is used for multiple L2 VNs w/ distributed L3 gateway function on NVEs (shown in Figure 4(a)), or L2 VNs and L3 VNs interconnected w/ distributed L3 gateway function on NVEs (shown in Figure 4(b)).

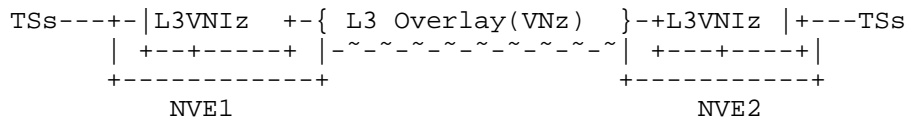
In the former case, the VAP may be VLAN or port/vport based. Tenant systems on the same L2 VN or different L2 VNs may be on the same or different NVEs. The tenant systems on the same L2 VN are in a broadcast domain and can be communicated without a constraint. The implementation looks like L2 NVE described above. For the traffic across L2 VNs, i.e. from one L2 VN to another, the tenant system sends the packet with GW MAC address that is configured on the local



NVE. The same MAC address should be used on all NVEs. (Use of different GW MAC address on NVEs is for further study). The NVE performs the IP table lookup and gets the destination VN, and gets the destination MAC address, and outer address or VAP address in the VN, then replace VN ID and destination MAC address on the packet, and further adds the outer addresses on the packet if sending to a peer NVE or send to the VAP where destination tenant system attaches to. L2 overlay is used between NVEs (NVE get the inner/outer address mapping from NVA). Operator can configure the communication policy between L2 VNs not only for unicast traffic but also broadcast/multicast traffic. The latter allows broadcast/multicast traffic sending to another L2 VN network. The policy can be associated to VNs instead of subnets or flows.

In the latter case shown in Figure 4(b), the L2 VN implementation on NVEs is like the L2 NVE described in section 4.1.1, the L3 VN implementation on NVEs is like the L3 NVE described in section 4.1.2. The distributed L3 gateway function on NVEs is used for local inter-VN traffic described above and policy be used for inter-VN traffic control. The NVEs in this case maintain both tenant system IP and MAC addresses.





(b)

Figure 4 L2/L3 NVE Service Model

The tenant network with multiple L2 VNs and L3 VNs interconnected w/distributed L3 gateway function MUST share the same IP and MAC address space. If the tenant network further interconnects with other tenant networks, external, or Internet via a gateway, either IP and/or MAC address partition or address translation MUST be used.

#### 4.2. Overlay Tunnel between NVEs

NVE may implement L2 overlay or L3 overlay depending on NVE types. A tunnel between two NVEs may be over one underlying network segment/domain or span across multiple network domains. Both NVEs need to use the same encapsulation protocol to encap/decap packets to/from a tunnel in between. There are several encapsulation methods in the industry such as VXLAN [VXLAN], NVGRE [NVGRE]. If two NVEs do not support the same encapsulation method, an interworking gateway is needed for the encapsulation translation. (It would be nice for DC operator and IETF to influence the industry to have one encapsulation protocol for the virtual network, which simplify the solution and lower its cost.) An NVE may support more than encapsulation methods, in this case, two NVEs need to select the same encapsulation methods. This can be done manually or via control plane negotiation. A P2MP tunnel or MP2MP tunnel may be used for multicast traffic transport. According to NVO3 architecture, an NVE relies on the NVA to obtain the inner/outer address mapping and the underlying network supports the IP connectivity between two NVEs. How NVA obtains the mapping information is outside the scope of the document.

The encapsulation process on an NVE also needs convey the packet characteristics in a VN to the underlying network, i.e. encode or translate the packet parameters in the inner header to the outer header, so that the packet can get the same treatment in the underlying network [NVO3DPREQ]. Some examples are CoS value, and entropy information calculation. The detail will be updated in next version.

#### 4.3. Multi-Tenancy Support

It is very important for NVO3 solution to support multi-tenancy over a common physical infrastructure, and ensures independent address space in individual tenant networks that are not communicated directly, i.e. only communicate with address translation or via Internet, and traffic isolation among them. NVE MUST maintain separated forwarding tables to support address overlapping. Since a tenant network may have one virtual or multiple virtual networks, it is important for a tenant or DC operator to manage the address allocation for the virtual networks in a tenant network to avoid address collision. A tunnel between NVEs may carry the traffic belonging to different virtual networks. The VN ID in the overlay header serves the traffic segregation.

#### 4.4. Route Path Control

A tenant network may be implemented as a full mesh among NVEs and not have a policy on route path at all. As the result, single hop is between sender TS and destination TS. A tenant network may contain some tenant systems that are designated as a network service appliance. In the case, tenant may want some tenant traffic passing through some network service appliance prior to delivering to the destination tenant and some are not. For example, Tenant network is implemented with three VNs in a DC. One L2 VN is for Web Tier, second L2 VN is for application tier; third L2 VN is for Database Tier. The policies are illustrated in the following figure. Traffic from the Web Tier to the App Tier MUST pass through tenant firewall on the Tenant System, say A; The traffic from App Tier to Web Tier can be directly routed; Traffic between App Tier and DB Tier MUST pass tenant firewall on tenant system, say B. No communication is allowed between the Web Tier and the DB Tier.

```

Web Tier ----FW A----> App Tier ----FW B----> DB Tier
<-----              <----FW B-----

```

Figure 5 Policies on a Three-Tier Tenant Network

In this case, An NVE attached by the firewall tenant system A is configured to forward all the traffic from Web Tier to the system A via a VAP. The system A processes the packets and may forward the packets to the App Tier via another VAP that is associated to App Tier on the NVE. The NVE forwards to the tenant systems in App Tier on behalf of the system A. The traffic from App Tier can be forwarded to the Web Tier directly. The NVE simply obtains the inner/outer address mapping and translate VN IDs on the packets prior to forwarding to the peer NVEs. Thus, in the pass-thru

firewall case, the inner/outer address mapping that an NVE gets from NVA is not destination tenant address (inner)/its NVE address; the outer address is the address of the NVE which the system A attaches to. The NVO3 architecture, such route path control can be implemented in NVA, NVE, or both. How to implement such policies on NVE is either beyond the scope of the document or for further study.

#### 4.5. Split-NVE

Split-NVE may be used in several use cases [NVO3ARCH]. Some NVE functions may reside on NVE spoke and some are on NVE hub. An overlay tunnel is used between NVE spoke and hub. One useful splitting structure in the data plane is to simplify the forwarding table on the NVE spoke, i.e. only maintains local forwarding entry; and let the NVE hub maintain the complete forwarding table. An NVE spoke just sends the packets to the NVE hub if the receiving point is not at local. It is possible that an NVE hub does not have any direct attached TS but connects to many NVE spokes. In this case, all the NVE spokes and NVE hubs are the member of one VN. Another useful structure is Intra NVE and Inter NVE splitting. The Intra NVEs forwards the packets if the sender TS and receiver TS are in the same VN and forwards the packets to the Inter NVE if not. The Inter NVE forwards the packets between the different VNs. The Inter NVE is often called a gateway. Note that, this design may cause the packet hire-pinning if the sender and receiver TSs in two different VNs are on the same Intra NVE. See section 5.3.

Split-NVE applies to all NVE service types. There is no configuration and behavior change between a TS and attaching NVE regardless if split-NVE is used or not. However, the communication performance and the tenant network cost may differ. The splitting control plane functionality on an NVE is outside the scope of this document.

#### 4.6. Multi-Homing Support

Two multi-homing of NVEs scenarios are described in NVO3 architecture document [NVO3ARCH]. 1) One NVE may have more than one overlay paths in term of more than one reachable IP addresses. 2) When an NVE is physically separated from attached tenant systems, a tenant system may attach to one or more NVEs via the VAPs.

For L2 NVE service, with the help of control plane protocols (e.g., mcLAG, ICCP, MP-BGP, etc), it can provide active/active or active/standby multi-homing access for tenant systems. For active/active mode, two-way traffics between tenant systems and multiple NVEs can be forwarded by per-flow or per-vlan load-

balancing. NVA may provide more than one inner/outer mapping to an NVE, the NVE may support some ECMP capability to distribute the traffic among the paths.

For L3 NVE service, NVEs are the first hop router for local tenant systems regardless of inter-subnet or intra-subnet traffic. The link/node redundancy mechanisms (e.g., ECMP, VRRP, etc) can provide various modes (i.e., active/active, active/standby) of multi-homing access for tenant systems.

For L2/3 NVE service, the main extension to above 2 types is the internal distributed gateway function on NVEs. Since the gateway function is used for process the ingress traffic on individual NVEs, multi-homing implementation is the same as of L2 NVE or L3 NVE service type.

#### 4.7. OAM Tools on NVE

It is necessary for an NVE to support some OAM tools. A tool can be turned on when a tunnel or VN is set up or dynamically turned on/off according to operation needs. The NVE implementation SHALL fulfill the OAM requirements described in [NVO3OAM]. Memo: followings are in considerations. This section is for the future study.

OAM tools on NVE should be operated under the conditions:

- . Run various OAM tools along the same path as data frames of overlay network between a pair of NVEs
- . Run OAM tools between per-tenant NVEs to probe the status of tunnel or NVE entities;
- . Send fault notification from underlay network to overlay network for its fault handing and alarm suppression.

NVE may support following tools but not limit to:

- . Connectivity Fault Detection: detect the tunnel connectivity fault between two or more NVEs that support the same virtual network;
- . Overlay Path Traceroute: trace the overlay path hops between two NVEs;
- . Underlying Path Traceroute: trace the underlay path hops between two NVEs;

- . Performance Monitoring: monitor various performance metrics such as packet loss, packet delay, packet delay variation, packet throughput, etc.
- . NVE Auto Discovery: dynamically discover other NVEs that support the same virtual network;
- . Send fault notification from underlay network to overlay network for its fault handling and alarm suppression.

## 5. Operation Considerations

### 5.1. VM Mobility

VM Mobility provides some benefits for DC operator in term of resource optimizations and performance turning. If a tenant system on a VM runs guest OS and application software, it can be moved from one NVE to another NVE without the impact of the live application. If the tenant systems running a network service appliances software, it may not easily move from one NVE to another NVE because there are some special configuration and policy on the NVE and tenant system to make it work. (See section 3.2) Furthermore if a VN is implemented with a special route graph instead of the "full" mesh route topology among NVEs, moving a VM from one NVE to another may require the route graph change, which may require some configuration changes on NVEs.

A VN ID is used to segregate the traffic for different VNs in data plane, or say on "wire". NVE implementation may use a domain-wide global VN ID or egress NVE assigned local VN ID in the data plane. If use of local VN ID, when a VM moves from one NVE to another, a sender NVE not only has to obtain the new NVE address the VM moves to, i.e. the outer addresses, also has to obtain the new VN ID the new NVE allocating for the VN. In other words, the ingress NVE has to modify both overlay header and outer header when a VM moves from one NVE to another. If a domain-wide VN ID is used on NVEs, an ingress NVE only need to modify the outer header when a VN moves. Although local allocated VN ID adds implementation complexity for VM mobility, it has advantage in associating VN ID with other context at egress NVE to facilitate egress NVE packet processing. Thus, either should be allowed for different use cases.

### 5.2. Gateway vs. Distributed Gateway

A gateway is used, in general, to interconnect two networks. A gateway can be implemented on a physical device, a bare metal server, or a VM on a server. The gateway in NVO3 means a virtual network

overlay interconnecting with other networks. The other network can be a physical network, a virtual network, a virtual network overlay, or Internet, which is often called an external network. Distributed gateway is a gateway function that is implemented on the NVEs so the traffic between two tenant systems in different virtual networks can be routed on the local NVE directly. The main benefit to use the distributed gateway is path optimization, which is important for some cloud applications and underlying networks.

To interconnect two networks, a gateway may integrate with other network service appliances such as NAT, Firewall and handle policy based forwarding. This means that inter-VN traffic subjects to this additional treatment. Note that a tenant often uses this as a rule in an application networking design. When implementing distributed gateway, that means that NVEs also need support such capability, which, sometimes, may become complex.

Should a tenant uses a gateway or distributed gateway for two VN interconnection? Here are the general recommendations.

- . If a VN overlay interconnects to an external network that is a physical network, virtual network, or Internet, using a gateway is practical. In this case, it is easy to place a gateway on the traffic path.
- . If a VN overlay in a DC interconnects to another VN overlay in another DC, the inter-VN traffic will pass through DC GWs, i.e. traffic pattern is north-south, it is good to use a gateway.
- . If a VN overlay interconnects to another VN overlay within a DC, i.e. traffic pattern is east-west, and there is light network service appliance function and/or policies for inter-VN traffic, using distributed gateway is better; otherwise use a gateway.

An L2 VN overlay may interconnect with another VN overlay and also interconnect with WAN or Internet. In this case, the L3 distributed gateway on the NVE not only be used for the VN interconnections within the NVEs but also be a member of an L3 overlay which the gateway connecting to WAN or Internet is the member of. Figure 6 illustrates an example. In this example, there are three virtual networks: VN<sub>x</sub>, VN<sub>i</sub>, and VN<sub>y</sub>. The VN<sub>x</sub> and VN<sub>x</sub> are the L2 overlay. The VN<sub>i</sub> is the L3 overlay. The VN<sub>x</sub> presents on NVE1 and an L2GW device. The L2GW device is also on a VLAN<sub>a</sub> which some metal servers attach to. Thus the tenant systems belonging to the VN<sub>x</sub> and the metal servers on VLAN<sub>a</sub> are on one L2 virtual network. (This is a useful configuration during the DC migration.) The VN<sub>y</sub> has the tenant systems on NVE1 too. NVE1 has L3dGW function. The inter VN traffic

on the NVE1 are passed through the L3dGW. The VNi terminates on the L3dGW at NVE1 and a vGW at the DCGW. NVEs get the vGW information from NVA. The tenant systems on NVE1 can reach the WAN via VNi, i.e. the L3dGW on NVE1 interconnects three VNs. The vGW on the DCGW terminates the traffic from VNi and passes to the WAN or vice versa. Note that some policy may be applied at the L3dGW to control the inter-VN traffic route. For instance, the VNx can exchange traffic with the VNy and the WAN; the VNy can exchange the traffic with the VNx only. Other policies and/or network appliances can be placed at the DCGW and associated with the vGW on the DCGW.

A gateway and distributed gateway function on NVEs can further work together to provide the inter-VN connection. This is useful when some NVEs in a VN can support distributed gateway function and others can't. A VN can have both distributed gateway on some NVEs and a gateway, which means that the traffic between two tenant systems may be forwarded locally if the attached NVE supports the distributed gateway function or may be hire-pin via a gateway if local NVE does not support distributed gateway function. Figure 6 illustrates this scenario too. The L2GW and NVE2 are attached metal servers and TSs, respectively. The NVE2 does not have distributed gateway capability. The L2 Overlay (VNx) exists between L2GW and DCGW and terminates on the L2VNix of the L2GW and the vGW on DCGW, respectively. Similar case for the L2 Overlay VNy is showed between NVE2 and DCGW. The gateway and distributed gateway combination usage can be very useful in the migration time.

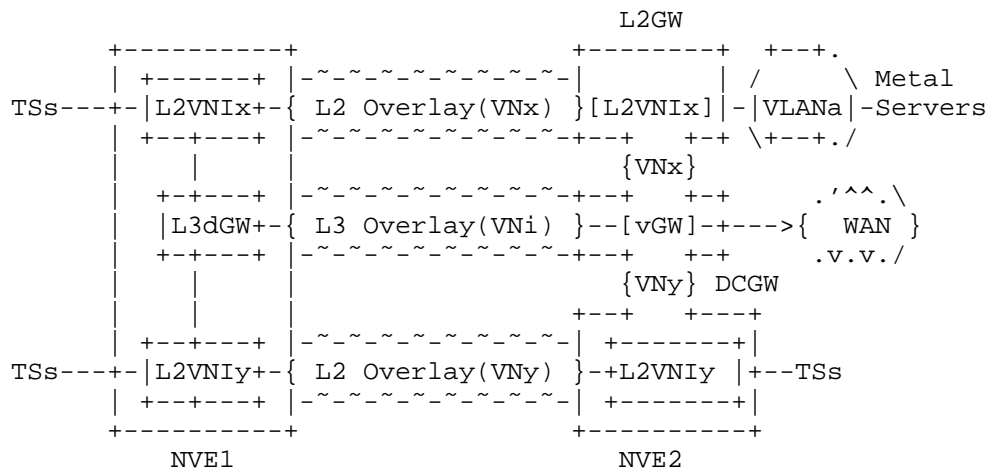




Figure 6 Example of Gateways and Distributed GW Usage

## 6. Security Considerations

NVO3 networks may be deployed in various use cases [NVO3CASE]. Different cases may have different level of security requirements for NVO3 networks. NVE is a key element for the security of NVO3 network. NVE should support the mutual or automated authentication with NVAs, other NVEs and tenant systems, to guarantee its peer having valid identity and privilege to communicate. NVEs should also provide integrity, confidentiality, and origin Authentication protection for whether control or data traffic against the unsecure underlay network. A per-tunnel based signatures or digests may provide data origin authentication, non-repudiation, and integrity protection. In addition, an NVE itself need to tolerant the DoS attack.

In the Split-NVE case, there are security risks that the NVE may be polluted by a compromised hypervisor with incorrect network updating information. However in this circumstance, the security damages can be limited to the hypervisor and the VNs attached to the compromised hypervisor. There are still ways to protect the attached NVE itself and mitigate the damages.

When an NVE is in the hypervisor, there are additional security risks on the NVE if the hypervisor may be compromised. A compromised NVE may send data traffic of a VN which it is not supposed to send. It is very important for an NVE to prevent any security risk initiated from a compromised remote NVE. The NVE may use the inner-to-outer address mappings table to filter incoming data traffic to ensure the inner address sourced packet originated from a correct participating NVE address.

If the tenant traffic privacy is a concern, cryptographic measures must be applied in addition. Confidentiality and integrity on the tenant data plane traffic could avoid the tenant traffic to be redirected, intercepted or modified by a compromised underlay network component.

In additional, the NVE implementation shall fulfill the security requirements described in [nvo3-security-requirements].

## 7. Acknowledgements

Authors like to thank Qin Wu for the review and valuable comments.

## 8. IANA Considerations

The document does not require any IANA action.

## 9. References

### 9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC2119, March 1997.

[RFC4834] Morin, T., "Requirements for Multicast in Layer 3 Provider-Provisioned Virtual Private Networks (PPVPNs)", RFC4834, April 2007

### 9.2. Informative References

[NVO3ARCH] Black, D., Narten, T., et al, "An Architecture for Overlay Networks (NVO3)", draft-narten-nvo3-arch, work in progress.

[NVO3CASE] Yong, L., et al, "Use Cases for DC Network Virtualization Overlays", draft-ietf-nvo3-use-case, work in progress

[NVO3DPREQ] Bitar, N., et al, "NVO3 Data Plane Requirements", draft-ietf-nvo3-dataplane-requirements-01, work in progress

[NVO3FRWK] LASSERRE, M., Motin, T., et al, "Framework for DC Network Virtualization", draft-ietf-nvo3-framework, work in progress.

[NVO3OAM] Ashwood, P, et al, "NVO3 Operation Requirement", draft-ashwood-nvo3-operational-requirement, work in progress

[NVGRE] Sridharan, M., et al, "NVGRE: Network Virtualization using Generic Routing Encapsulation", draft-sridharan-virtualization-nvgre, work in progress

[VXLAN] Mahalingam, M., Dutt, D., etc, "VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", draft-mahalingam-dutt-dcops-vxlan, work in progress

### Authors' Addresses

Lucy Yong  
Huawei Technologies, USA

Email: lucy.yong@huawei.com

Frank Liang Xia  
Huawei Technologies

Email: frank.xialiang@huawei.com

Qiang Zu  
Ericsson  
Email: zu.qiang@ericsson.com

